

Kind Inference for Datatypes: Technical Supplement

NINGNING XIE, The University of Hong Kong, China
 RICHARD A. EISENBERG, Bryn Mawr College, USA and Tweak I/O
 BRUNO C. D. S. OLIVEIRA, The University of Hong Kong, China

ACM Reference Format:

Ningning Xie, Richard A. Eisenberg, and Bruno C. d. S. Oliveira. 2020. Kind Inference for Datatypes: Technical Supplement. *Proc. ACM Program. Lang.* 4, POPL, Article 53 (January 2020), 88 pages. <https://doi.org/10.1145/3371121>

This technical supplement to *Kind Inference for Datatypes* serves to expand upon the text in the main paper. It contains detailed typing rules, proofs, and connections to the Glasgow Haskell Compiler (GHC). Sections in this document are meant to connect to sections in the main paper. There are many hyperlinks throughout, especially those highlighting the connections to GHC; you may wish to read on a computer instead of on paper.

A OTHER LANGUAGE EXTENSIONS

This section accompanies Section 8 of the main paper, including discussion about more related language extensions. These extensions affect kind inference, but not in a fundamental way.

A.1 Visible Dependent Quantification

Besides specified type variables for which users can optionally provide type arguments, Haskell also incorporates *visible dependent quantification* (VDQ)¹, e.g., `type T :: ∀(k :: ★) → k → ★`, with which users are forced to provide type arguments to *T*. That is, one would use *T* with, e.g., `T ★ Int` and `T (★ → ★) Maybe`, never just `T Int`. Visible dependent quantification is Haskell's equivalent to routine dependent quantification in dependently typed languages.

To support VDQ, rule **DT-TT** needs to be extended, as VDQ brings variables into scope for later reference. For example, given

```
type T :: ∀(k :: ★) → k → ★
data T k a = MkT
```

We should get a context `k :: ★, a :: k` when checking `MkT`.

VDQ opens an interesting design choice: should unannotated type variables be able to introduce VDQ? For example, in the definition of *P* below, we use *f* and *a* as the arguments to *T*. To make it type-check, we need to infer `P :: ∀(f :: ★) → f → ★`.

¹In GHC 8.8, GHC infers kinds using VDQ, but users are not allowed to write VDQ explicitly. This has been rectified for the GHC 8.10 release, as described in [this proposal](#).

Authors' addresses: Ningning Xie, The University of Hong Kong, Department of Computer Science, Hong Kong, China, nxxie@cs.hku.hk; Richard A. Eisenberg, Bryn Mawr College, Department of Computer Science, Bryn Mawr, PA, USA, Tweak I/O, rae@richarde.dev; Bruno C. d. S. Oliveira, The University of Hong Kong, Department of Computer Science, Hong Kong, China, bruno@cs.hku.hk.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

© 2020 Copyright held by the owner/author(s).
 2475-1421/2020/1-ART53
<https://doi.org/10.1145/3371121>

data $P f a = MkP (T f a)$

However, the tricky part with inferring the kind of P is that we cannot have a fixed initial form of the kind of P , i.e., $\widehat{\alpha} \rightarrow \widehat{\beta} \rightarrow \star$ or $\forall(f : \widehat{\alpha}) \rightarrow \widehat{\beta} \rightarrow \star$, when type-checking the `rec` group of P , until we type-check P 's body. In order to avoid this challenge, we support GHC's current ruling on the matter: *dependent variables must be manifestly so*. That is, the initial kind of a datatype includes VDQ only for those variables that appear, lexically, in the kind of a variable; other type parameters are reflected in a datatype's initial kind with a regular (non-dependent) arrow. This guideline rejects P as an example of non-manifest dependency.

A.2 Datatype Promotion

Haskellers can use datatypes as kinds and can write data constructors in types [Yorgey et al. 2012]. In the PolyKinds system, types and kinds are mixed (allowing datatypes to be used as kinds), but there is no facility to use a data constructor in a type.

To support such usage, the kinding judgment must now use the term context to fetch the type of data constructors. Moreover, dependency analysis needs to take dependencies on data constructors into account.

Definition A.1 (Dependency Analysis with Type-Level Data). *We extend Definition 6.1 with*

(iii) *The definition of T_1 depends on the definition of T_2 if T_1 uses data constructors of T_2 .*

While the appearance of data constructors in types enriches the type language considerably, they do not pose a particular challenge for inference; the rest of our presentation would remain unaffected.

A.3 Partial Type Signatures

For quite some time, GHC has supported kind signatures on a subset of a datatype's parameters, much like the partial type signatures described by ?. For example, `App`, below, does not have a signature but still has a kind annotation for f .

data $App (f :: \star \rightarrow \star) a = A (f a)$

To deal with such a construct we first need to amend the syntax of a datatype declaration to support kind annotations for variables.

$$\text{datatype decl. } \mathcal{T} ::= \text{data } T \phi = \overline{\mathcal{D}_j^j}$$

Kind annotations can also contain free variables, which need to be generalized in a similar way as signatures. For example, `T2` has kind $\forall\{k :: \star\}. \forall(f :: k). \star$.

data $T2 (f :: k) = MkT2$

Supporting these partial signatures adds complication to rule `PGM-DT-TT` (and its algorithmic counterpart) to bring the kind variables into scope. However, and critically, a partial signature will still go via rule `PGM-DT-TT`, never rule `PGM-DT-TTS`, used for full signatures only. This means that a partial type signature does *not* unlock polymorphic recursion: the datatype will be considered monomorphic and ungeneralized within its own recursive group.

B TODAY'S GHC

This paper describes, in depth, how kind inference can work for datatype declarations. Here, we review how our work relates to GHC. To make the claims concrete, this section contains references to specific stretches of code within GHC.

B.1 Constraint-Based Type Inference

Type inference in GHC is based on generating and solving constraints [Pottier and Rémy 2005; Vytiniotis et al. 2011], distinct from our approach here, where we unify on the fly. Despite this different architecture, our results carry over to the constraint-based style. Instead of using eager unification, we can imagine accumulating constraints in output contexts Θ , and then invoking a solver to extend the context with solutions. This approach is taken by Eisenberg [2016].

In thinking about the change from eager unification to delayed constraints, one might worry about information loss around any place where we apply a context as a substitution, as these substitutions would be empty in a constraint-solving approach without eager unification. At top-level (Figure 6), a constraint-solving approach would run the constraint solver, and the substitutions would contain the same mappings as our approach provides. Conversely, the relations in Figure 8 would become part of the constraint solver, so substituting here is safe, too. A potential problem arises in rule `A-KTT-APP` (Figure 7), where we substitute in the function's kind before running the kind-directed \Vdash^{kapp} judgment. However, our system is predicative: it never unifies a type variable with a polytype. Thus, the substitution in rule `A-KTT-APP` can never trigger a new usage of rule `A-KAPP-TT-FORALL`. It *can* distinguish between rule `A-KAPP-TT-ARROW` and rule `A-KAPP-TT-KUVAR`, but we conjecture that the choice between these rules is irrelevant: both will lead to equivalent substitutions in the end.

B.2 Contexts

A typing context is *not* maintained in much of GHC's inference algorithm. Instead, a variable's kind is stored in the `data structure` representing the variable. This is very convenient, as it means that looking up a variable's type or kind is a pure, fast operation. One downside is that the compiler must maintain an extra invariant that all occurrences of a variable store the same kind; this is straightforward to maintain in practice.

Beyond just storing variables' kinds, the typing context in this paper also critically stores variables' ordering. Lacking contexts, GHC uses a different mechanism: *level numbers*, *originally invented* to implement untouchability [Vytiniotis et al. 2011, Section 5.1]. Every type variable in GHC is assigned a level number during inference. Type variables contain a `structure` that includes level numbers. Roughly, the level number of a type variable a corresponds to the number of type variables in scope before a . Accordingly, we can tell the relative order (in a hypothetical context, according to the systems in this paper) of two variables simply by comparing their level numbers. One of GHC's *invariants* is that a unification variable at level n is never unified with a type that mentions a variable with a level number $m > n$; this is much like the extra checks in the unification judgments in our paper.

The *local scopes* of this paper are also *tracked* by GHC. All the variables in the same local scope are assigned the same level number, and they are flagged as reorderable. After inference is complete, *GHC does a topological sort* to get the final order.

A final role that contexts play in our formalism is that they store solutions for unification variables; we apply contexts as a substitution. In GHC, unification variables store mutable cells that get filled in. It has a process called *zonking*,² which is exactly analogous to our use of contexts as substitutions. Zonking a unification variable replaces the variable with its solution, if any.

²There are actually two variants of zonking in GHC: we can zonk *during type-checking* or *at the end*. The difference between the variants is chiefly what to do for an unfilled unification variable. The former leaves them alone, while the latter has to default them somehow; details are beyond our scope here.

B.3 Unification

The solver in GHC still has to carry out unification, much along the lines of the unification judgment we present here. This algorithm has to deal with the heterogeneous unification problems we consider, as well. Indeed, GHC’s unification algorithm recurs into the kinds of a unification variable and the type it is unifying with, just as ours does. As implied by our focus on decidability of unification, there have been a number of bugs in GHC’s implementation that led to loops in the type checker; the most recent is [#16902](#).

GHC actually uses several unification algorithms internally. It has an eager unifier, much like the one we describe. When that unifier fails, it generates the constraint that is sent to the solver. (The eager unifier is meant solely to be an optimization.) There is also a unifier meant to work after type inference is complete; it checks for instance overlap, for example. All the unifiers recur into kinds:

- [The eager unifier recurs into kinds.](#)
- [The unifier in the solver recurs into kinds.](#)
- The pure unifier uses an invariant that the kinds are related before looking at the types. [It must recur when decomposing applications.](#)

In addition, GHC also has an overlap problem within unification, as exhibited in our paper by the overlap between rules [A-U-KVARL](#) and [A-U-KVARR](#) in Figure 3. Both [the eager unifier](#) and [the constraint-solver unifier](#) deal with this ambiguity by using heuristics to choose which variable might be more suitable for unification. This particular issue—which variable to unify when there is a choice—has been the subject of some amount of churn over the years.

B.4 Promotion

The promotion operation, too, is present in GHC, though its form is quite different than what we have presented. Instead of promoting during unification, GHC simply refuses to solve a unification variable if any of the free variables of its supposed solution lives to the right of the variable in the context. Because GHC is working with constraints, it just leaves the unification problem as an unsolved constraint. If there remain unsolved constraints, GHC then [promotes](#) the variables it can: some cannot be promoted because they depend on locally bound quantified (not unification) type variables.

B.5 Complete User-Supplied Kinds

Along with stand-alone kind signatures, as described in this paper, GHC supports *complete user-supplied kinds*, or CUSKs. A datatype has a CUSK when [certain syntactic conditions](#) are satisfied; GHC detects these conditions *before* doing any kind inference. These CUSKs are a poor substitute for proper kind signatures, as the syntactic cues are fragile and unexpected: users sometimes write a CUSK without meaning to, and also sometimes leave out a necessary part of a CUSK when they intend to specify the kind. Stand-alone kind signatures are a new feature; they begin with the keyword **type** instead of **data**, as we have used in our paper.

Interestingly, it would be wrong to support CUSKs in a system without polymorphic kinds. Consider this example:

```
data S1 a = MkT1 S2
data S2 = MkS2 (S1 Maybe)
```

The types *S1* and *S2* form a group. We put *S2* (which has a CUSK) into the context with kind \star . When we check *S1*, we find no constraints on *a* (in the constraint-generation pass; see the general approach below). The kind of *S1* is then defaulted to $\star \rightarrow \star$. Checking *S2* fails. Instead, we wish to

pretend that $S2$ does not have a CUSK. This would mean that constraint-generation happens for all the constructors in both $S1$ and $S2$, and $S1$ would get its correct kind $(\star \rightarrow \star) \rightarrow \star$.

With kind-polymorphism, we have no problem because the kind of $T1$ will be generalized to $\forall(k :: \star). k \rightarrow \star$.

This was reported as bug [#16609](#).

B.6 Dependency Analysis

The algorithm implemented in GHC for processing datatype declarations starts with dependency analysis, as ours does. The dependency analysis is less fine-grained than what we have proposed in this paper: signatures are ignored in the dependency analysis, and so datatypes with signatures are processed alongside all the others. This means that the kinds in the example below have more restrictive kinds in GHC than they do in our system:

```
data S1 ::  $\forall k. k \rightarrow \star$ 
data S1 a = MkS1 (S2 Int)
data S2 a = MkS2 (S3 Int)
data S3 a = MkS3 (S1 Int)
```

A naïve dependency analysis would put all three definitions in the same group. The kind for $S1$ is given; it would indeed have that kind. The parameters of $S2$ and $S3$ would initially have an unknown kind, but when occurrences of $S2$ and $S3$ are processed (in the definitions of $S1$ and $S2$, respectively), this unknown kind would become \star . Neither $S2$ nor $S3$ would be generalized.

There is a ticket to improve the dependency analysis: [#9427](#).

B.7 Approach to Kind-Checking Datatypes

GHC's approach is summarized in [this comment](#). Overall kind-checking is orchestrated by [this function](#).

After dependency analysis, so-called *initial kinds* are produced for all the datatypes in the group. These either come from a datatype's CUSK or from a simple analysis of the header of the datatype (without looking at constructors). This step corresponds to our algorithm's placing a binding for the datatype in the context, either with the kind signature or with a unification variable (rules [A-PGM-DT-TTS](#) and [A-PGM-DT-TT](#)).

If there is no CUSK, GHC then [passes](#) over all the datatype's constructors, collecting constraints on unification variables. After solving these constraints, GHC [generalizes](#) the datatype kind.

For all datatypes, now with generalized kinds, all data constructors are checked (again, for non-CUSK types). Because the kinds of the types are now generalized, [this pass](#) infers any invisible parameters to polykinded types. For non-CUSK types, this second pass using generalized kinds replaces the $T_i \mapsto T_i @\phi_i^c$ substitution in the context in the last premise to rule [A-PGM-DT-TT](#). Performing a substitution—instead of re-generating and solving constraints—may be an opportunity for improvement in GHC.

B.8 Syntax for GADTs

Haskell's *syntax* for GADT declarations is very troublesome. Consider these examples:

```
data R a where
  MkR :: b  $\rightarrow$  R b

data S a where
  MkS :: S b

data T a where
  MkT ::  $\forall(k :: \star) (b :: k). T b$ 
```

In GHC’s implementation of GADTs, any variables declared in the header (between **data** and **where**) *do not scope*. In all the examples above, the type variable a does not scope over the constructor declarations. This is why we have written the variable b in those types, to make it clear that b is distinct from a . We could have written a —it would still be a distinct a from that in the header—but it would be more confusing.

The question is: how do we determine the kind of the parameter to the datatype? One possibility is to look only in the header. In all cases above, we would infer no constraints and would give each type a kind of $\forall(k :: \star). k \rightarrow \star$. This is unfortunate, as it would make R a kind-indexed GADT: the MkR constructor would carry a proof that the kind of its type parameter is \star . This, in turn, wreaks havoc with type inference, as it is hard to infer the result type of a pattern-match against a GADT [Vytiniotis et al. 2011].

Furthermore, this approach might accept *more* programs than the user wants. Consider this definition:

data P a **where**

$MkP1 :: b \rightarrow P\ b$

$MkP2 :: f\ a \rightarrow P\ f$

Does the user want a kind-indexed GADT, noting that b and f have different kinds? Or would the user want this rejected? If we make the fully general kind $\forall k. k \rightarrow \star$ for P , this would be accepted, perhaps surprising users.

It thus seems we wish to look at the data constructors when inferring the kind of the datatype. The challenge in looking at data constructors is that their variables are *locally* bound. In MkR and MkS , we implicitly quantify over b . In MkR , we discover that $b :: \star$, and thus that R must have kind $\star \rightarrow \star$. In MkS , we find no constraints on b ’s kind, and thus no constraints on S ’s argument’s kind, and so we can generalize to get $S :: \forall(k :: \star). k \rightarrow \star$. Let us now examine MkT : it explicitly brings k and b into scope. Thus, the argument to T has *local* kind k . It would be impossible to unify the kind of T ’s argument—call it \hat{a} —with k , because k would be bound to the *right* of \hat{a} in an inference context. Thus it seems we would reject T .

This result is also dissatisfying. In practice, GHC implements an ad-hoc algorithm, described in Section B.9.

Our conclusion here is that the design of GADTs in GHC/Haskell is flawed: the type variables mentioned in the header should indeed scope over the constructors. This would mean we could reject T : if the user wanted to explicitly make T polykinded, they could do so right in the header. We recognize that it would be hard to make this change today, but one result of this work is the interplay between scoping (order in the context) and unification; the current state of affairs will always require ad-hoc support.

B.9 Polymorphic Recursion

One challenge in kind inference is in the handling of polymorphic recursion. Although non-CUSK types are indeed monomorphic during the constraint-generation pass, some limited form of polymorphic recursion can get through. This is because all type variables are represented by a special form of unification variable called a TyVarTv. TyVarTvs can unify only with other type variables. This design is motivated by the following examples:

data $T1$ ($a :: k$) $b = MkT1$ ($T2\ a\ b$)

data $T2$ ($c :: j$) $d = MkT2$ ($T1\ c\ d$)

data $T3$ a **where**

$MkT3 :: \forall(k :: \star) (b :: k). T3\ b$

We want to accept all of these definitions. The first two, $T1$ and $T2$, form a mutually recursive group. Neither has a CUSK. However, the recursive occurrences are not polymorphically recursive: both recursive occurrences are at the *same* kind as the definition. Yet the first parameter to $T1$ is declared to have kind k while the first parameter to $T2$ is declared to have kind j . The solution: allow k to unify with j during the constraint-generation pass. We would *not* want to allow either k or j to unify with a non-variable, as that would seem to go against the user's wishes. But they must be allowed to unify with each other to accept this example.

With $T3$ (identical to T from Section B.8), we have a different motivation. During inference, we will guess the kind of a ; call it $\hat{\alpha}$. When checking the $MkT3$ constructor, we will need to unify $\hat{\alpha}$ with the locally bound k . We cannot set $\hat{\alpha} := k$, as that will fill $\hat{\alpha}$ with a k , bound to $\hat{\alpha}$'s *right* in the context. Instead, we must set $k := \hat{\alpha}$. This is possible only if k is represented by a unification variable.

There are two known problems with this approach:

- (1) It sometimes accepts polymorphic recursion, even without a CUSK. Here is an example:

```
data T4 a =  $\forall(k :: \star)$  (b :: k). MkT4 (T4 b)
```

The definition of $T4$ is polymorphically recursive: the occurrence $T4 b$ is specialized to a kind other than the kind of a . Yet this definition is accepted. The two kinds unify (as k becomes a unification variable, set to the guessed kind of a) during the constraint-generation pass. Then, $T4$ is generalized to get the kind $\forall k. k \rightarrow \star$, at which point the last pass goes through without a hitch.

The reason this acceptance is troublesome is not that $T4$ is somehow dangerous or unsafe. It is that we know that polymorphic recursion cannot be inferred [Henglein 1993], and yet GHC does it. Invariably, this must mean that GHC's algorithm will be hard to specify beyond its implementation.

This wrinkle is described [on the GHC wiki](#).

- (2) In rare cases, the constraint-generation pass will succeed, while the final pass—meant to be redundant—will fail. Here is an example:

```
data SameKind :: k  $\rightarrow$  k  $\rightarrow$  Type
```

```
data Bad a where
```

```
  MkBad ::  $\forall k_1 k_2$  (a :: k1) (b :: k2). Bad (SameKind a b)
```

During the constraint-generation pass, the kinds k_1 and k_2 are allowed to unify, accepting the definition of Bad . During the final pass, however, k_1 and k_2 are proper quantified type variables, always distinct. Thus the $SameKind a b$ type is ill-kinded and rejected.

The fact that this final pass can fail means that we cannot implement it via a simple substitution, as we do in rule A-PGM-DT-TT. One possible solution is our suggestion to change the scoping of type parameters to GADT-syntax datatype declarations. With that change, our second motivation above for TyVarTvs would disappear. GHC could then use TyVarTvs only for kind variables in the head of a datatype declaration, using proper quantified type variables in constructors. Of course, this change would break much code in the wild, and we do not truly expect it to ever be adopted.

This problem is documented in [this comment](#).

B.10 The Quantification Check

Our quantification check (Section 7.2) also has a [parallel in GHC](#), but GHC's solution to the problem differs from ours. Instead of rejecting programs that fail the quantification check, GHC accepts them, replacing the variables that would be (but cannot be) quantified with its constant $Any :: \forall k. k$. The Any type is uninhabited, but exists at all kinds. As such, it is an appropriate replacement

for unquantifiable, unconstrained unification variables. Yet this decision in GHC has unfortunate consequences: the *Any* type can appear in error messages, and its introduction induces hard-to-understand type errors.

The GHC developers are questioning their approach to this problem. See [this comment](#) and [this ticket](#).

Another design alternative is to generalize the variable to the leftmost position where it is still well-formed. Recall the example in Section 7.2:

```
data Proxy :: ∀k. k → ★
data Relate :: ∀a (b :: a). a → Proxy b → ★
data T :: ∀(a :: ★) (b :: a) (c :: a) d. Relate b d → ★
```

We have $d :: \widehat{\alpha}$, and $\widehat{\alpha} = \text{Proxy } \widehat{\beta}$, with $\widehat{\beta} :: a$. As there are no further constraints on $\widehat{\beta}$, the definition of T is rejected by the quantification check.

Instead of rejecting the program, or solving $\widehat{\beta}$ using *Any*, we can generalize over $\widehat{\beta}$ as a fresh variable f , which is put after a to make it well-kinded. Namely, we get

```
data T :: ∀(a :: ★) {f :: a} (b :: a) (c :: a) (d :: Proxy f). Relate @a @f b d → ★
```

However, this ordering of the variables violates our declarative specification. Moreover, this type requires an inferred variable to be between specified variables. With higher-rank polymorphism, due to the fact that GHC does not support first-class type-level abstraction (i.e., Λ in types), this type cannot be instantiated to

```
∀(a :: ★) (b :: a) (c :: a) (d :: Proxy f). Relate @a @b b d → ★
```

or

```
∀(a :: ★) (b :: a) (c :: a) (d :: Proxy f). Relate @a @c b d → ★
```

which makes the generalization less useful.

B.11 ScopedSort

When GHC deals with a local scope—a set of variables that may be reordered—it does a topological sort on the variables at the end. However, not any topological sort will do: it must use one that preserves the left-to-right ordering of the variables as much as possible. This is because GHC considers these implicitly bound variables to be *specified*: they are available for visible type application. For example, recall the example from Section 2.2, modified slightly:

```
data Q (a :: (f b)) (c :: k) (x :: f c)
```

Inference will tell us that k must come before f and b , but the order of f and b is immaterial. Our approach here is to make f , b , and k *inferred* variables: users of Q will not be able to instantiate these parameters with visible type application. However, GHC takes a different view: because the user has written the names of f , b , and k , they will be *specified*. This choice means that the precise sorting algorithm GHC uses to fix the order of local scopes becomes part of the *specification* of the language. Indeed, GHC documents the precise algorithm in its [manual](#). If we followed suit, the algorithm would have to appear in our declarative specification, which goes against the philosophy of a declarative system.

Some recent debate led to a [conclusion](#) that we would change the interpretation of the Q example from the main paper, meaning that its kind variables would indeed become *inferred*. However, the problem with ScopedSort still exists in type signatures, where type variables may be implicitly bound.

B.12 The “Forall-or-Nothing” Rule

GHC implements the so-called *forall-or-nothing rule*, which states that either *all* variables are quantified by a user-written `forall`, or none are. These examples illustrate the effect:

ex1 :: $a \rightarrow b \rightarrow a$
ex2 :: $\forall a b. a \rightarrow b \rightarrow a$
ex3 :: $\forall a. a \rightarrow b \rightarrow a$
ex4 :: $(\forall a. a \rightarrow b \rightarrow a)$

The signatures for both *ex1* and *ex2* are accepted: *ex1* quantifies none, while *ex2* quantifies all. The signature for *ex3* is rejected, as GHC rejects a mixed economy. However, and perhaps surprisingly, *ex4* is accepted. The only difference between *ex3* and *ex4* is the seemingly-redundant parentheses. However, because the forall-or-nothing rule applies only at the top level of a signature, the rule is not in effect for the \forall in *ex4*.

This rule interacts with the main paper only in that our formalism (and some of our examples) does not respect it. This may be the cause of differing behavior between GHC and the examples we present.

C COMPLETE SET OF RULES

In this section we include the complete set of rules. Some of the rules are repeated from those in the paper.

C.1 Declarative Haskell98

$$\boxed{\Sigma \vdash^k \sigma : \kappa} \quad \text{(Kinding for Polymorphic Types)}$$

$$\frac{\text{K-FORALL} \quad \Sigma, a : \kappa \vdash^k \sigma : \star}{\Sigma \vdash^k \forall a : \kappa. \sigma : \star}$$

$$\boxed{\Sigma \vdash \Psi} \quad \text{(Well-formed Term Contexts)}$$

$$\frac{\text{ECTX-EMPTY} \quad \Sigma \vdash \bullet}{\Sigma \vdash \bullet} \quad \text{ECTX-DCON} \quad \frac{\Sigma \vdash \Psi \quad \Sigma \vdash^k \sigma : \star}{\Sigma \vdash \Psi, D : \sigma}$$

C.2 Algorithmic Haskell98

$$\boxed{\Delta \Vdash^k \sigma : \kappa \dashv \Theta} \quad \text{(Kinding for Polymorphic Types)}$$

$$\frac{\text{A-K-FORALL} \quad \Delta \Vdash^{kv} \kappa \quad \Delta, a : \kappa \Vdash^k \sigma : \kappa_2 \dashv \Theta, a : \kappa \quad [\Theta]\kappa_2 = \star}{\Delta \Vdash^k \forall a : \kappa. \sigma : \star \dashv \Theta}$$

$$\boxed{\Delta \Vdash^{kc} \sigma \Leftarrow \kappa} \quad \text{(Checking)}$$

$$\frac{\text{A-KC-EQ} \quad \Delta \Vdash^k \sigma : \kappa_1 \dashv \Delta \quad [\Delta]\kappa_1 = [\Delta]\kappa_2}{\Delta \Vdash^{kc} \sigma \Leftarrow \kappa_2}$$

$$\boxed{\Delta \Vdash^{kv} \kappa} \quad \text{(Well-formed Kinds)}$$

$$\frac{\text{A-KV-STAR} \quad \Delta \Vdash^{kv} \star}{\Delta \Vdash^{kv} \star} \quad \text{A-KV-ARROW} \quad \frac{\Delta \Vdash^{kv} \kappa_1 \quad \Delta \Vdash^{kv} \kappa_2}{\Delta \Vdash^{kv} \kappa_1 \rightarrow \kappa_2} \quad \text{A-KV-KUVAR} \quad \frac{\widehat{\alpha} \in \Delta}{\Delta \Vdash^{kv} \widehat{\alpha}}$$

$$\boxed{\Delta \text{ ok}} \quad \text{(Well-formed Type Contexts)}$$

$\frac{\text{A-TCTX-EMPTY}}{\bullet \text{ ok}}$	$\frac{\text{A-TCTX-TVAR} \quad \Delta \text{ ok} \quad \Delta \Vdash^{\text{kv}} \kappa}{\Delta, a : \kappa \text{ ok}}$	$\frac{\text{A-TCTX-TCON} \quad \Delta \text{ ok} \quad \Delta \Vdash^{\text{kv}} \kappa}{\Delta, T : \kappa \text{ ok}}$	$\frac{\text{A-TCTX-KUVAR} \quad \Delta \text{ ok}}{\Delta, \widehat{\alpha} \text{ ok}}$
$\frac{\text{A-TCTX-KUVAR}^{\text{SOLVED}} \quad \Delta \text{ ok} \quad \Delta \Vdash^{\text{kv}} \kappa}{\Delta, \widehat{\alpha} = \kappa \text{ ok}}$			

$$\boxed{\Delta \Vdash^{\text{ectx}} \Gamma} \quad \text{(Well-formed Term Contexts)}$$

$\frac{\text{A-ECTX-EMPTY}}{\Delta \Vdash^{\text{ectx}} \bullet}$	$\frac{\text{A-ECTX-DCON} \quad \Delta \Vdash^{\text{ectx}} \Gamma \quad \Delta \Vdash^{\text{kc}} \sigma \Leftarrow \star}{\Delta \Vdash^{\text{ectx}} \Gamma, D : \sigma}$
---	--

$$\boxed{\Delta \longrightarrow \Omega} \quad \text{(Defaulting)}$$

$\frac{\text{A-CTXDE-EMPTY}}{\bullet \longrightarrow \bullet}$	$\frac{\text{A-CTXDE-TVAR} \quad \Delta \longrightarrow \Omega}{\Delta, a : \kappa \longrightarrow \Omega, a : \kappa}$	$\frac{\text{A-CTXDE-TCON} \quad \Delta \longrightarrow \Omega}{\Delta, T : \kappa \longrightarrow \Omega, T : \kappa}$	$\frac{\text{A-CTXDE-KUVAR}^{\text{SOLVED}} \quad \Delta \longrightarrow \Omega}{\Delta, \widehat{\alpha} = \kappa \longrightarrow \Omega, \widehat{\alpha} = \kappa}$
$\frac{\text{A-CTXDE-SOLVE} \quad \Delta \longrightarrow \Omega}{\Delta, \widehat{\alpha} \longrightarrow \Omega, \widehat{\alpha} = \star}$			

C.3 Context Application in Haskell98

$[\Delta]\kappa \text{ applies } \Delta \text{ as a substitution to } \kappa.$ $[\Delta]\star = \star$ $[\Delta]\kappa_1 \rightarrow \kappa_2 = [\Delta]\kappa_1 \rightarrow [\Delta]\kappa_2$ $[\Delta][\widehat{\alpha}]\widehat{\alpha} = \widehat{\alpha}$ $[\Delta][\widehat{\alpha} = \kappa]\widehat{\alpha} = [\Delta][\widehat{\alpha} = \kappa]\kappa$	$[\Delta]\Gamma \text{ applies } \Delta \text{ as a substitution to } \Gamma.$ $[\Delta]\bullet = \bullet$ $[\Delta](\Gamma, D : \sigma) = [\Delta]\Gamma, D : [\Delta]\sigma$
$[\Omega]\Delta \text{ applies } \Omega \text{ as a substitution to } \Delta.$ $[\bullet]\bullet = \bullet$ $[\Omega, a : \kappa](\Delta, a : \kappa) = [\Omega]\Delta, a : [\Omega]\kappa$ $[\Omega, T : \kappa](\Delta, T : \kappa) = [\Omega]\Delta, T : [\Omega]\kappa$ $[\Omega, \widehat{\alpha} = \kappa](\Delta, \widehat{\alpha}) = [\Omega]\Delta$ $[\Omega, \widehat{\alpha} = \kappa](\Delta, \widehat{\alpha} = \kappa') = [\Omega]\Delta \quad \text{if } [\Omega]\kappa = [\Omega]\kappa'$ $[\Omega, \widehat{\alpha} = \kappa]\Delta = [\Omega]\Delta \quad \text{if } \widehat{\alpha} \notin \Delta$	

C.4 Context Extension in Haskell98

$$\boxed{\Delta \longrightarrow \Theta} \quad \text{(Context Extension)}$$

$\frac{\text{A-CTXE-EMPTY}}{\bullet \longrightarrow \bullet}$	$\frac{\text{A-CTXE-TVAR} \quad \Delta \longrightarrow \Theta}{\Delta, a : \kappa \longrightarrow \Theta, a : \kappa}$	$\frac{\text{A-CTXE-TCON} \quad \Delta \longrightarrow \Theta}{\Delta, T : \kappa \longrightarrow \Theta, T : \kappa}$	$\frac{\text{A-CTXE-KUVAR} \quad \Delta \longrightarrow \Theta}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha}}$
$\frac{\text{A-CTXE-KUVAR}^{\text{SOLVED}} \quad \Delta \longrightarrow \Theta \quad [\Theta]\kappa_1 = [\Theta]\kappa_2}{\Delta, \widehat{\alpha} = \kappa_1 \longrightarrow \Theta, \widehat{\alpha} = \kappa_2}$	$\frac{\text{A-CTXE-SOLVE} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha} = \kappa}$	$\frac{\text{A-CTXE-ADD} \quad \Delta \longrightarrow \Theta}{\Delta \longrightarrow \Theta, \widehat{\alpha}}$	$\frac{\text{A-CTXE-ADD}^{\text{SOLVED}} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta \longrightarrow \Theta, \widehat{\alpha} = \kappa}$

C.5 Declarative PolyKinds

 $\boxed{\lceil \sigma \rceil}$ (Kind results in \star)

$$\frac{\text{SR-STAR}}{\lceil \star \rceil}$$

$$\frac{\text{SR-ARROW} \quad \lceil \kappa_2 \rceil}{\lceil \kappa_1 \rightarrow \kappa_2 \rceil}$$

$$\frac{\text{SR-FORALL} \quad \lceil \sigma \rceil}{\lceil \forall \phi. \sigma \rceil}$$

 $\boxed{\Sigma \vdash^{\text{inst}} \mu_1 : \eta \sqsubseteq \omega \rightsquigarrow \mu_2}$

(Instantiation)

INST-REFL

$$\frac{}{\Sigma \vdash^{\text{inst}} \mu : \omega \sqsubseteq \omega \rightsquigarrow \mu}$$

INST-FORALL

$$\frac{\Sigma \vdash^{\text{ela}} \rho : \omega_1 \quad \Sigma \vdash^{\text{inst}} \mu_1 @ \rho : \eta[a \mapsto \rho] \sqsubseteq \omega_2 \rightsquigarrow \mu_2}{\Sigma \vdash^{\text{inst}} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2}$$

INST-FORALL-INFER

$$\frac{\Sigma \vdash^{\text{ela}} \rho : \omega_1 \quad \Sigma \vdash^{\text{inst}} \mu_1 @ \rho : \eta[a \mapsto \rho] \sqsubseteq \omega_2 \rightsquigarrow \mu_2}{\Sigma \vdash^{\text{inst}} \mu_1 : \forall \{a : \omega_1\}. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2}$$

 $\boxed{\Sigma \vdash^{\text{kc}} \sigma \Leftarrow \omega \rightsquigarrow \mu}$

(Kind Checking)

KC-SUB

$$\frac{\Sigma \vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu_1 \quad \Sigma \vdash^{\text{inst}} \mu_1 : \eta \sqsubseteq \omega \rightsquigarrow \mu_2}{\Sigma \vdash^{\text{kc}} \sigma \Leftarrow \omega \rightsquigarrow \mu_2}$$

 $\boxed{\Sigma \vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu}$

(Kinding)

KTT-STAR

$$\frac{}{\Sigma \vdash^{\text{k}} \star : \star \rightsquigarrow \star}$$

KTT-NAT

$$\frac{}{\Sigma \vdash^{\text{k}} \text{Int} : \star \rightsquigarrow \text{Int}}$$

KTT-VAR

$$\frac{(a : \omega) \in \Sigma}{\Sigma \vdash^{\text{k}} a : \omega \rightsquigarrow a}$$

KTT-ARROW

$$\frac{}{\Sigma \vdash^{\text{k}} \rightarrow : \star \rightarrow \star \rightarrow \star \rightsquigarrow \rightarrow}$$

KTT-TCON

$$\frac{(T : \eta) \in \Sigma}{\Sigma \vdash^{\text{k}} T : \eta \rightsquigarrow T}$$

KTT-APP

$$\frac{\Sigma \vdash^{\text{k}} \tau_1 : \eta_1 \rightsquigarrow \rho_1 \quad \Sigma \vdash^{\text{inst}} \rho_1 : \eta_1 \sqsubseteq (\omega_1 \rightarrow \omega_2) \rightsquigarrow \rho_2 \quad \Sigma \vdash^{\text{kc}} \tau_2 \Leftarrow \omega_1 \rightsquigarrow \rho_3}{\Sigma \vdash^{\text{k}} \tau_1 \tau_2 : \omega_2 \rightsquigarrow \rho_2 \rho_3}$$

KTT-KAPP

$$\frac{\Sigma \vdash^{\text{k}} \kappa_1 : \forall a : \omega. \eta \rightsquigarrow \rho_1 \quad \Sigma \vdash^{\text{kc}} \kappa_2 \Leftarrow \omega \rightsquigarrow \rho_2}{\Sigma \vdash^{\text{k}} \kappa_1 @ \kappa_2 : \eta[a \mapsto \rho_2] \rightsquigarrow \rho_1 @ \rho_2}$$

KTT-KAPP-INFER

$$\frac{\Sigma \vdash^{\text{k}} \kappa_1 : \forall \{\overline{a_i} : \omega_i^i\}. \forall a : \omega. \eta \rightsquigarrow \rho'_1}{\Sigma \vdash^{\text{ela}} \rho_i : \omega_i[\overline{a_i} \mapsto \rho_i^i] \quad \Sigma \vdash^{\text{kc}} \kappa_2 \Leftarrow \omega[\overline{a_i} \mapsto \rho_i^i] \rightsquigarrow \rho'_2}$$

$$\frac{}{\Sigma \vdash^{\text{k}} \kappa_1 @ \kappa_2 : \eta[\overline{a_i} \mapsto \rho_i^i][a \mapsto \rho_2] \rightsquigarrow \rho'_1 @ \rho_i^i @ \rho'_2}$$

KTT-FORALL

$$\frac{\Sigma \vdash^{\text{kc}} \kappa \Leftarrow \star \rightsquigarrow \omega \quad \Sigma, a : \omega \vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow \mu}{\Sigma \vdash^{\text{k}} \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega. \mu}$$

KTT-FORALLI

$$\frac{\Sigma \vdash^{\text{ela}} \omega : \star \quad \Sigma, a : \omega \vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow \mu}{\Sigma \vdash^{\text{k}} \forall a. \sigma : \star \rightsquigarrow \forall a : \omega. \mu}$$

 $\boxed{\Sigma \vdash^{\text{ela}} \mu : \eta}$

(Elaborated Kinding)

ELA-STAR

$$\frac{}{\Sigma \vdash^{\text{ela}} \star : \star}$$

ELA-NAT

$$\frac{}{\Sigma \vdash^{\text{ela}} \text{Int} : \star}$$

ELA-VAR

$$\frac{(a : \omega) \in \Sigma}{\Sigma \vdash^{\text{ela}} a : \omega}$$

ELA-TCON

$$\frac{(T : \eta) \in \Sigma}{\Sigma \vdash^{\text{ela}} T : \eta}$$

ELA-ARROW

$$\frac{}{\Sigma \vdash^{\text{ela}} \rightarrow : \star \rightarrow \star \rightarrow \star}$$

ELA-APP

$$\frac{\Sigma \vdash^{\text{ela}} \rho_1 : \omega_1 \rightarrow \omega_2 \quad \Sigma \vdash^{\text{ela}} \rho_2 : \omega_1}{\Sigma \vdash^{\text{ela}} \rho_1 \rho_2 : \omega_2}$$

ELA-KAPP

$$\frac{\Sigma \vdash^{\text{ela}} \rho_1 : \forall a : \omega. \eta \quad \Sigma \vdash^{\text{ela}} \rho_2 : \omega}{\Sigma \vdash^{\text{ela}} \rho_1 @ \rho_2 : \eta[a \mapsto \rho_2]}$$

$$\begin{array}{c}
\text{ELA-KAPP-INFER} \\
\frac{\Sigma \vdash^{\text{ela}} \rho_1 : \forall \{a : \omega\}. \eta \quad \Sigma \vdash^{\text{ela}} \rho_2 : \omega}{\Sigma \vdash^{\text{ela}} \rho_1 @ \rho_2 : \eta[a \mapsto \rho_2]} \\
\text{ELA-FORALL} \\
\frac{\Sigma \vdash^{\text{ela}} \omega : \star \quad \Sigma, a : \omega \vdash^{\text{ela}} \mu : \star}{\Sigma \vdash^{\text{ela}} \forall a : \omega. \mu : \star} \\
\text{ELA-FORALL-INFER} \\
\frac{\Sigma \vdash^{\text{ela}} \omega : \star \quad \Sigma, a : \omega \vdash^{\text{ela}} \mu : \star}{\Sigma \vdash^{\text{ela}} \forall \{a : \omega\}. \mu : \star}
\end{array}$$

 $\Sigma \text{ ok}$ *(Well-formed Type Contexts)*

$$\begin{array}{c}
\text{TCTX-EMPTY} \\
\frac{}{\bullet \text{ ok}} \\
\text{TCTX-TVAR-TT} \\
\frac{\Sigma \text{ ok} \quad \Sigma \vdash^{\text{ela}} \rho : \star}{\Sigma, a : \rho \text{ ok}} \\
\text{TCTX-TCON-TT} \\
\frac{\Sigma \text{ ok} \quad \Sigma \vdash^{\text{ela}} \eta : \star}{\Sigma, T : \eta \text{ ok}}
\end{array}$$

 $\Sigma \vdash \Psi$ *(Well-formed Term Contexts)*

$$\begin{array}{c}
\text{ECTX-EMPTY} \\
\frac{}{\Sigma \vdash \bullet} \\
\text{ECTX-DCON-TT} \\
\frac{\Sigma \vdash \Psi \quad \Sigma \vdash^{\text{ela}} \mu : \star}{\Sigma \vdash \Psi, D : \mu}
\end{array}$$

C.6 Algorithmic PolyKinds

 $\Delta \Vdash^{\text{inst}} \mu_1 : \eta \sqsubseteq \omega \rightsquigarrow \mu_2 \dashv \Theta$ *(Instantiation)*

$$\begin{array}{c}
\text{A-INST-REFL} \\
\frac{\Delta \Vdash^{\text{u}} \omega_1 \approx \omega_2 \dashv \Theta}{\Delta \Vdash^{\text{inst}} \mu : \omega_1 \sqsubseteq \omega_2 \rightsquigarrow \mu \dashv \Theta} \\
\text{A-INST-FORALL} \\
\frac{\Delta, \hat{\alpha} : \omega_1 \Vdash^{\text{inst}} \mu_1 @ \hat{\alpha} : \eta[a \mapsto \hat{\alpha}] \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}{\Delta \Vdash^{\text{inst}} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta} \\
\text{A-INST-FORALL-INFER} \\
\frac{\Delta, \hat{\alpha} : \omega_1 \Vdash^{\text{inst}} \mu_1 @ \hat{\alpha} : \eta[a \mapsto \hat{\alpha}] \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}{\Delta \Vdash^{\text{inst}} \mu_1 : \forall \{a : \omega_1\}. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}
\end{array}$$

 $\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \eta \rightsquigarrow \mu \dashv \Theta$ *(Kind Checking)*

$$\begin{array}{c}
\text{A-KC-SUB} \\
\frac{\Delta \Vdash^{\text{kc}} \sigma : \eta \rightsquigarrow \mu_1 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\text{inst}} \mu_1 : [\Delta_1] \eta \sqsubseteq [\Delta_1] \omega \rightsquigarrow \mu_2 \dashv \Delta_2}{\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \omega \rightsquigarrow \mu_2 \dashv \Delta_2}
\end{array}$$

 $\Delta \Vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu \dashv \Theta$ *(Kinding)*

$$\begin{array}{c}
\text{A-KTT-STAR} \\
\frac{}{\Delta \Vdash^{\text{k}} \star : \star \rightsquigarrow \star \dashv \Delta} \\
\text{A-KTT-NAT} \\
\frac{}{\Delta \Vdash^{\text{k}} \text{Int} : \star \rightsquigarrow \text{Int} \dashv \Delta} \\
\text{A-KTT-VAR} \\
\frac{(a : \omega) \in \Delta}{\Delta \Vdash^{\text{k}} a : \omega \rightsquigarrow a \dashv \Delta} \\
\text{A-KTT-TCON} \\
\frac{(T : \eta) \in \Delta}{\Delta \Vdash^{\text{k}} T : \eta \rightsquigarrow T \dashv \Delta} \\
\text{A-KTT-ARROW} \\
\frac{}{\Delta \Vdash^{\text{k}} \rightarrow : \star \rightarrow \star \rightarrow \star \rightsquigarrow \rightarrow \dashv \Delta} \\
\text{A-KTT-FORALL} \\
\frac{\Delta \Vdash^{\text{kc}} \kappa \Leftarrow \star \rightsquigarrow \omega \dashv \Delta_1 \quad \Delta_1, a : \omega \Vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow \mu \dashv \Delta_2, a : \omega, \Delta_3 \quad \Delta_3 \hookrightarrow a}{\Delta \Vdash^{\text{k}} \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega. [\Delta_3] \mu \dashv \Delta_2, \text{unsolved}(\Delta_3)} \\
\text{A-KTT-APP} \\
\frac{\Delta \Vdash^{\text{k}} \tau_1 : \eta_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\text{kapp}} (\rho_1 : [\Delta_1] \eta_1) \bullet \tau_2 : \omega \rightsquigarrow \rho \dashv \Theta}{\Delta \Vdash^{\text{k}} \tau_1 \tau_2 : \omega \rightsquigarrow \rho \dashv \Theta} \\
\text{A-KTT-FORALLI} \\
\frac{\Delta, \hat{a} : \star, a : \hat{a} \Vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow \mu \dashv \Delta_2, a : \hat{a}, \Delta_3 \quad \Delta_3 \hookrightarrow a}{\Delta \Vdash^{\text{k}} \forall a. \sigma : \star \rightsquigarrow \forall a : \hat{a}. [\Delta_3] \mu \dashv \Delta_2, \text{unsolved}(\Delta_3)}
\end{array}$$

$$\begin{array}{c}
\text{A-KTT-KAPP} \\
\frac{\Delta \Vdash^k \tau_1 : \eta \rightsquigarrow \rho_1 + \Delta_1 \quad [\Delta_1]\eta = \forall a : \omega. \eta_2 \quad \Delta_1 \Vdash^k \tau_2 \Leftarrow \omega \rightsquigarrow \rho_2 + \Delta_2}{\Delta \Vdash^k \tau_1 @ \tau_2 : \eta_2[a \mapsto \rho_2] \rightsquigarrow \rho_1 @ \rho_2 + \Delta_2} \\
\text{A-KTT-KAPP-INFER} \\
\frac{\Delta \Vdash^k \tau_1 : \eta \rightsquigarrow \rho_1 + \Delta_1 \quad \Delta_1, \widehat{\alpha}_i : \omega_i[\overline{a_i \mapsto \widehat{\alpha}_i^i}] \Vdash^k \tau_2 \Leftarrow \omega[\overline{a_i \mapsto \widehat{\alpha}_i^i}] \rightsquigarrow \rho_2 + \Delta_2}{\Delta \Vdash^k \tau_1 @ \tau_2 : \eta_2[\overline{a_i \mapsto \widehat{\alpha}_i^i}][a \mapsto \rho_2] \rightsquigarrow \rho_1 @ \widehat{\alpha}_i^i @ \rho_2 + \Delta_2}
\end{array}$$

$$\boxed{\Delta \Vdash^{\text{kapp}} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 + \Theta} \quad (\text{Application Kinding})$$

$$\begin{array}{c}
\text{A-KAPP-TT-ARROW} \\
\frac{\Delta \Vdash^k \tau \Leftarrow \omega_1 \rightsquigarrow \rho_2 + \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \omega_1 \rightarrow \omega_2) \bullet \tau : \omega_2 \rightsquigarrow \rho_1 \rho_2 + \Theta} \\
\text{A-KAPP-TT-FORALL} \\
\frac{\Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{kapp}} (\rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}]) \bullet \tau : \omega \rightsquigarrow \rho + \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \forall a : \omega_1. \eta) \bullet \tau : \omega \rightsquigarrow \rho + \Theta} \\
\text{A-KAPP-TT-FORALL-INFER} \\
\frac{\Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{kapp}} (\rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}]) \bullet \tau : \omega \rightsquigarrow \rho + \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \forall \{a : \omega_1\}. \eta) \bullet \tau : \omega \rightsquigarrow \rho + \Theta} \\
\text{A-KAPP-TT-KUVAR} \\
\frac{\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \Vdash^k \tau \Leftarrow \widehat{\alpha}_1 \rightsquigarrow \rho_2 + \Theta}{\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash^{\text{kapp}} (\rho_1 : \widehat{\alpha}) \bullet \tau : \widehat{\alpha}_2 \rightsquigarrow \rho_1 \rho_2 + \Theta}
\end{array}$$

$$\boxed{\Delta \Vdash^{\text{ela}} \mu : \eta} \quad (\text{Elaborated Kinding})$$

$$\begin{array}{c}
\text{A-ELA-STAR} \quad \text{A-ELA-KUVAR} \quad \text{A-ELA-NAT} \quad \text{A-ELA-VAR} \quad \text{A-ELA-TCON} \\
\frac{}{\Delta \Vdash^{\text{ela}} \star : \star} \quad \frac{(\widehat{\alpha} : \omega) \in \Delta}{\Delta \Vdash^{\text{ela}} \widehat{\alpha} : [\Delta]\omega} \quad \frac{}{\Delta \Vdash^{\text{ela}} \text{Int} : \star} \quad \frac{(a : \omega) \in \Delta}{\Delta \Vdash^{\text{ela}} a : [\Delta]\omega} \quad \frac{(T : \eta) \in \Delta}{\Delta \Vdash^{\text{ela}} T : [\Delta]\eta} \\
\text{A-ELA-ARROW} \quad \text{A-ELA-FORALL} \\
\frac{}{\Delta \Vdash^{\text{ela}} \rightarrow : \star \rightarrow \star \rightarrow \star} \quad \frac{\Delta \Vdash^{\text{ela}} \omega : \star \quad \Delta, a : \omega \Vdash^{\text{ela}} \mu : \star}{\Delta \Vdash^{\text{ela}} \forall a : \omega. \mu : \star} \\
\text{A-ELA-FORALL-INFER} \quad \text{A-ELA-APP} \\
\frac{\Delta \Vdash^{\text{ela}} \omega : \star \quad \Delta, a : \omega \Vdash^{\text{ela}} \mu : \star}{\Delta \Vdash^{\text{ela}} \forall \{a : \omega\}. \mu : \star} \quad \frac{\Delta \Vdash^{\text{ela}} \rho_1 : \omega_1 \rightarrow \omega_2 \quad \Delta \Vdash^{\text{ela}} \rho_2 : \omega_1}{\Delta \Vdash^{\text{ela}} \rho_1 \rho_2 : \omega_2} \\
\text{A-ELA-KAPP} \quad \text{A-ELA-KAPP-INFER} \\
\frac{\Delta \Vdash^{\text{ela}} \rho_1 : \forall a : \omega. \eta \quad \Delta \Vdash^{\text{ela}} \rho_2 : \omega}{\Delta \Vdash^{\text{ela}} \rho_1 @ \rho_2 : \eta[a \mapsto [\Delta]\rho_2]} \quad \frac{\Delta \Vdash^{\text{ela}} \rho_1 : \forall \{a : \omega\}. \eta \quad \Delta \Vdash^{\text{ela}} \rho_2 : \omega}{\Delta \Vdash^{\text{ela}} \rho_1 @ \rho_2 : \eta[a \mapsto [\Delta]\rho_2]}
\end{array}$$

$$\boxed{\Delta \Vdash_{\phi^c}^{\text{gen}} \Gamma_1 \rightsquigarrow \Gamma_2} \quad (\text{Generalization})$$

$$\begin{array}{c}
\text{A-GEN} \\
\frac{\widehat{\phi}_i^c = \text{unsolved}(\mu_i)}{\Delta \Vdash_{\phi^c}^{\text{gen}} \overline{D_i : \mu_i}^i \rightsquigarrow D : \forall \{\phi^c\}. \forall \{\phi_i^c\}. (\mu[\widehat{\phi}_i^c \mapsto \phi_i^c])}
\end{array}$$

$\Delta \text{ ok}$	<i>(Well-formed Type Contexts)</i>		
A-TCTX-EMPTY	A-TCTX-TVAR-TT $\Delta \text{ ok} \quad \Delta \Vdash^{\text{ela}} \omega : \star$	A-TCTX-TCON-TT $\Delta \text{ ok} \quad \Delta \Vdash^{\text{ela}} \eta : \star$	A-TCTX-KUVAR-TT $\Delta \text{ ok} \quad \Delta \Vdash^{\text{ela}} \omega : \star$
$\bullet \text{ ok}$	$\Delta, a : \omega \text{ ok}$	$\Delta, T : \eta \text{ ok}$	$\Delta, \widehat{\alpha} : \omega \text{ ok}$
	A-TCTX-KUVAR-SOLVED-TT $\Delta \text{ ok} \quad \Delta \Vdash^{\text{ela}} \omega_2 : [\Delta]\omega_1$	A-TCTX-LO $\Delta, \Delta^{\text{lo}} \text{ ok}$	A-TCTX-MARKER $\Delta \text{ ok}$
	$\Delta, \widehat{\alpha} : \omega_1 = \omega_2 \text{ ok}$	$\Delta, \{\Delta^{\text{lo}}\} \text{ ok}$	$\Delta, \blacktriangleright_D \text{ ok}$

$\Delta \Vdash^{\text{ectx}} \Gamma$	<i>(Well-formed Term Contexts)</i>		
	A-ECTX-EMPTY $\Delta \Vdash^{\text{ectx}} \bullet$	A-ECTX-DCON-TT $\Delta \Vdash^{\text{ectx}} \Gamma \quad \Delta \Vdash^{\text{ela}} \mu : \star$ $\Delta \Vdash^{\text{ectx}} \Gamma, D : \mu$	

$\Delta \Vdash^{\mu} \omega_1 \approx \omega_2 \vdash \Theta$	<i>(Unification)</i>		
A-U-REFL-TT $\Delta \Vdash^{\mu} \omega \approx \omega \vdash \Delta$	A-U-APP $\Delta \Vdash^{\mu} \rho_1 \approx \rho_3 \vdash \Delta_1 \quad \Delta_1 \Vdash^{\mu} [\Delta_1]\rho_2 \approx [\Delta_1]\rho_4 \vdash \Theta$ $\Delta \Vdash^{\mu} \rho_1 \rho_2 \approx \rho_3 \rho_4 \vdash \Theta$		
	A-U-KAPP $\Delta \Vdash^{\mu} \rho_1 \approx \rho_3 \vdash \Delta_1 \quad \Delta_1 \Vdash^{\mu} [\Delta_1]\rho_2 \approx [\Delta_1]\rho_4 \vdash \Theta$ $\Delta \Vdash^{\mu} \rho_1 @ \rho_2 \approx \rho_3 @ \rho_4 \vdash \Theta$		
A-U-KVARL-TT $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \vdash \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \quad \Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1 \Vdash^{\mu} [\Theta_1]\omega_1 \approx \omega_2 \vdash \Theta_3$	$\Delta \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \vdash \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2$		
A-U-KVARR-TT $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \vdash \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \quad \Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1 \Vdash^{\mu} [\Theta_1]\omega_1 \approx \omega_2 \vdash \Theta_3$	$\Delta \Vdash^{\mu} \rho_1 \approx \widehat{\alpha} \vdash \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2$		
A-U-KVARL-LO-TT $\Delta_1, \Delta_2 \Vdash^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \vdash \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4$ $\Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2]\omega_1 \approx \omega_2 \vdash \Theta_5, \{\Theta_6\}$	$\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \vdash \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$		
A-U-KVARR-LO-TT $\Delta_1, \Delta_2 \Vdash^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \vdash \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4$ $\Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2]\omega_1 \approx \omega_2 \vdash \Theta_5, \{\Theta_6\}$	$\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \rho_1 \approx \widehat{\alpha} \vdash \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$		

$\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \vdash \Theta$	<i>(Promotion)</i>		
A-PR-STAR $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \star \rightsquigarrow \star \vdash \Delta$	A-PR-ARR $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rightarrow \rightsquigarrow \rightarrow \vdash \Delta$	A-PR-TCON $\Delta[T][\widehat{\alpha}] \Vdash_{\widehat{\alpha}}^{\text{pr}} T \rightsquigarrow T \vdash \Delta[T][\widehat{\alpha}]$	
A-PR-NAT $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \text{Int} \rightsquigarrow \text{Int} \vdash \Delta$	A-PR-APP $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \rho_1 \vdash \Delta_1 \quad \Delta_1 \Vdash_{\widehat{\alpha}}^{\text{pr}} [\Delta_1]\omega_2 \rightsquigarrow \rho_2 \vdash \Theta$ $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \omega_2 \rightsquigarrow \rho_1 \rho_2 \vdash \Theta$		

$$\begin{array}{c}
\text{A-PR-KAPP} \\
\frac{\Delta \Vdash_{\widehat{\alpha}}^{\text{PR}} \omega_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash_{\widehat{\alpha}}^{\text{PR}} [\Delta_1]\omega_2 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta \Vdash_{\widehat{\alpha}}^{\text{PR}} \omega_1 @\omega_2 \rightsquigarrow \rho_1 @\rho_2 \dashv \Theta} \\
\text{A-PR-KUVARL} \\
\frac{}{\Delta[\widehat{\beta}][\widehat{\alpha}] \Vdash_{\widehat{\alpha}}^{\text{PR}} \widehat{\beta} \rightsquigarrow \widehat{\beta} \dashv \Delta[\widehat{\beta}][\widehat{\alpha}]} \\
\text{A-PR-KUVARR-TT} \\
\frac{\Delta \Vdash_{\widehat{\alpha}}^{\text{PR}} [\Delta]\rho \rightsquigarrow \rho_1 \dashv \Theta[\widehat{\alpha}][\widehat{\beta} : \rho]}{\Delta[\widehat{\alpha}][\widehat{\beta} : \rho] \Vdash_{\widehat{\alpha}}^{\text{PR}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_1 \dashv \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]} \\
\text{A-PR-TVAR} \\
\frac{}{\Delta[a][\widehat{\alpha}] \Vdash_{\widehat{\alpha}}^{\text{PR}} a \rightsquigarrow a \dashv \Delta[a][\widehat{\alpha}]} \\
\boxed{\Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow \Theta} \quad \text{(Moving)} \\
\text{A-MV-EMPTY} \\
\frac{}{\bullet \dashv^{\text{mv}} \Delta \rightsquigarrow \Theta} \\
\text{A-MV-KUVAR} \\
\frac{\text{vars}(\omega) \# \text{dom}(\Delta_2) \quad \Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow \Theta}{\widehat{\alpha} : \omega, \Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow \widehat{\alpha} : \omega, \Theta} \\
\text{A-MV-KUVARM} \\
\frac{\neg(\text{vars}(\omega) \# \text{dom}(\Delta_2)) \quad \Delta_1 \dashv^{\text{mv}} \Delta_2, \widehat{\alpha} : \omega \rightsquigarrow \Theta}{\widehat{\alpha} : \omega, \Delta_1 \dashv^{\text{mv}} \Delta \rightsquigarrow \Theta} \\
\text{A-MV-TVAR} \\
\frac{\text{vars}(\omega) \# \text{dom}(\Delta_2) \quad \Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow \Theta}{a : \omega, \Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow a : \omega, \Theta} \\
\text{A-MV-TVARM} \\
\frac{\neg(\text{vars}(\omega) \# \text{dom}(\Delta_2)) \quad \Delta_1 \dashv^{\text{mv}} \Delta_2, a : \omega \rightsquigarrow \Theta}{a : \omega, \Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow \Theta}
\end{array}$$

C.7 Context Application in PolyKinds

$$\begin{array}{l}
\text{[}\Delta\text{]}\eta \text{ applies } \Delta \text{ as a substitution to } \eta. \\
\begin{array}{l}
\text{[}\Delta\text{]}\star = \star \\
\text{[}\Delta\text{]}\text{Int} = \text{Int} \\
\text{[}\Delta\text{]}a = a \\
\text{[}\Delta\text{]}T = T \\
\text{[}\Delta\text{]}\rightarrow = \rightarrow \\
\text{[}\Delta\text{]}\forall a : \omega. \eta = \forall a : [\Delta]\omega. [\Delta]\eta \\
\text{[}\Delta\text{]}\forall \{a : \omega\}. \eta = \forall \{a : [\Delta]\omega\}. [\Delta]\eta \\
\text{[}\Delta\text{]}(\rho_1 \rho_2) = ([\Delta]\rho_1) ([\Delta]\rho_2) \\
\text{[}\Delta\text{]}(\rho_1 @\rho_2) = ([\Delta]\rho_1) @([\Delta]\rho_2) \\
\text{[}\Delta\text{]}[\widehat{\alpha}] = \widehat{\alpha} \\
\text{[}\Delta\text{]}[\widehat{\alpha} : \omega = \rho] = [\Delta\text{]}[\widehat{\alpha} : \omega = \rho]
\end{array} \\
\text{[}\Delta\text{]}\Gamma \text{ applies } \Delta \text{ as a substitution to } \Gamma. \\
\begin{array}{l}
\text{[}\Omega\text{]}\bullet = \bullet \\
\text{[}\Omega\text{]}(\Gamma, D : \mu) = [\Omega]\Gamma, D : [\Omega]\mu
\end{array}
\end{array}$$

$\text{[}\Omega\text{]}\Delta$ applies Ω as a substitution to Δ .

$$\begin{array}{l}
\text{[}\Omega\text{]}\bullet = \bullet \\
\text{[}\Omega, a : \omega\text{]}(\Delta, a : \omega) = [\Omega]\Delta, a : [\Omega]\omega \\
\text{[}\Omega, T : \omega\text{]}(\Delta, T : \omega) = [\Omega]\Delta, T : [\Omega]\omega \\
\text{[}\Omega, \widehat{\alpha} : \omega = \rho\text{]}(\Delta, \widehat{\alpha} : \omega) = [\Omega]\Delta \\
\text{[}\Omega, \widehat{\alpha} : \omega = \rho_1\text{]}(\Delta, \widehat{\alpha} : \omega = \rho_2) = [\Omega]\Delta \quad \text{if } [\Omega]\rho_1 = [\Omega]\rho_2 \\
\text{[}\Omega, \widehat{\alpha} : \omega = \rho\text{]}\Delta = [\Omega]\Delta \quad \text{if } \widehat{\alpha} \notin \Delta \\
\text{[}\Omega, \blacktriangleright_D\text{]}(\Delta, \blacktriangleright_D) = [\Omega]\Delta \\
\text{[}\Omega, \{\Omega_1\}\text{]}(\Delta, \{\Delta_1\}) = [\Omega, \Omega_1](\Delta, \Delta') \\
\text{where } \Delta' = \text{topo}(\Delta_1)
\end{array}$$

C.8 Context Extension in PolyKinds

$\Delta \longrightarrow \Theta$				<i>(Context Extension)</i>			
A-CTXE-EMPTY	A-CTXE-TVAR-TT $\Delta \longrightarrow \Theta$	A-CTXE-TCON-TT $\Delta \longrightarrow \Theta$	A-CTXE-KUVAR-TT $\Delta \longrightarrow \Theta$	A-CTXE-EMPTY	A-CTXE-TVAR-TT $\Delta, a : \omega \longrightarrow \Theta, a : \omega$	A-CTXE-TCON-TT $\Delta, T : \eta \longrightarrow \Theta, T : \eta$	A-CTXE-KUVAR-TT $\Delta, \widehat{\alpha} : \omega \longrightarrow \Theta, \widehat{\alpha} : \omega$
$\bullet \longrightarrow \bullet$	$\Delta, a : \omega \longrightarrow \Theta, a : \omega$	$\Delta, T : \eta \longrightarrow \Theta, T : \eta$	$\Delta, \widehat{\alpha} : \omega \longrightarrow \Theta, \widehat{\alpha} : \omega$	$\Delta \longrightarrow \Theta$	$\Theta \Vdash^{\text{ela}} \rho : [\Theta]\omega$	$\Delta \longrightarrow \Theta$	$\Theta \Vdash^{\text{ela}} \omega : \star$
A-CTXE-KUVAR-SOLVED-TT $\Delta \longrightarrow \Theta$	A-CTXE-SOLVE-TT $\Delta \longrightarrow \Theta$	A-CTXE-ADD-TT $\Delta \longrightarrow \Theta$	A-CTXE-ADD-SOLVED-TT $\Delta \longrightarrow \Theta$	A-CTXE-MARKER $\Delta \longrightarrow \Theta$	A-CTXE-LO $\Delta \longrightarrow \Theta$	A-CTXE-ADD-TT $\Delta \longrightarrow \Theta$	A-CTXE-ADD-TT $\Delta \longrightarrow \Theta$
$\Delta, \widehat{\alpha} : \omega = \rho_1 \longrightarrow \Theta, \widehat{\alpha} : \omega = \rho_2$	$\Delta, \widehat{\alpha} : \omega \longrightarrow \Theta, \widehat{\alpha} : \omega = \rho$	$\Delta \longrightarrow \Theta, \Theta \Vdash^{\text{ela}} \rho : [\Theta]\omega$	$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega$	$\Delta, \text{topo}(\Delta_1) \longrightarrow \Theta, \Theta_1$	$\Delta, \{\Delta_1\} \longrightarrow \Theta, \{\Theta_1\}$	$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega$	$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega$
$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega = \rho$	$\Delta, \blacktriangleright_D \longrightarrow \Theta, \blacktriangleright_D$	$\Delta \longrightarrow \Theta, \Theta \Vdash^{\text{ela}} \rho : [\Theta]\omega$	$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega$	$\Delta, \text{topo}(\Delta_1) \longrightarrow \Theta, \Theta_1$	$\Delta, \{\Delta_1\} \longrightarrow \Theta, \{\Theta_1\}$	$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega$	$\Delta \longrightarrow \Theta, \widehat{\alpha} : \omega$

D PROOF FOR HASKELL98

D.1 List of Lemmas

D.1.1 Well-formedness of Declarative Type System.

Lemma D.1 (Well-formedness of Declarative Typing Data Constructor Declaration). *If $\Sigma \Vdash_{\tau_1}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau_2$, then $\Sigma \Vdash^{\text{k}} \tau_2 : \star$.*

Lemma D.2 (Well-formedness of Declarative Typing Datatype Declaration). *If $\Sigma \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, then $\Sigma \vdash \Psi$.*

D.1.2 Well-formedness of Algorithmic Type System.

Lemma D.3 (Well-formedness of Promotion). *If $\Delta_1, \widehat{\alpha}, \Delta_2 \text{ ok}$, and $\Delta_1, \widehat{\alpha}, \Delta_2 \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa_1 \rightsquigarrow \kappa_2 \dashv \Theta$, then $\Theta = \Theta_1, \widehat{\alpha}, \Theta_2$, and $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta$, and $\Theta_1 \Vdash^{\text{kv}} \kappa_2$, and $\Theta \text{ ok}$. By weakening, there is also $\Theta \Vdash^{\text{kv}} \kappa_2$.*

Lemma D.4 (Well-formedness of Unification). *If $\Delta \text{ ok}$, and $\Delta \Vdash^{\text{u}} \kappa_1 \approx \kappa_2 \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \text{ ok}$.*

Lemma D.5 (Well-formedness of Application Kinding). *If $\Delta \text{ ok}$, and $\Delta \Vdash^{\text{kapp}} \kappa_1 \bullet \kappa_2 : \kappa \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \text{ ok}$. Moreover, if $\Delta \Vdash^{\text{kv}} \kappa_1$, then we have $\Theta \Vdash^{\text{kv}} \kappa$.*

Lemma D.6 (Well-formedness of Kinding). *If $\Delta \text{ ok}$, and $\Delta \Vdash^{\text{k}} \sigma : \kappa \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \text{ ok}$, and $\Theta \Vdash^{\text{kv}} \kappa$ and $\Theta \Vdash^{\text{k}} \sigma \Leftarrow \star$.*

Lemma D.7 (Well-formedness of Typing Data Constructor Declarations). *If $\Delta \text{ ok}$, and $\Delta \Vdash_{\tau'}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \text{ ok}$. and $\Theta \Vdash^{\text{k}} \tau \Leftarrow \star$.*

Lemma D.8 (Well-formedness of Typing Datatype Declaration). *If $\Delta \text{ ok}$, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \text{ ok}$, and $\Theta \Vdash^{\text{ectx}} \Gamma$.*

D.1.3 Properties of Context Extension.

Lemma D.9 (Declaration Preservation). *If $\Delta \longrightarrow \Theta$, if a type constructor or a type variable or a kind unification variable is declared in Δ , then it is declared in Θ .*

Lemma D.10 (Extension Weakening). *Given $\Delta \longrightarrow \Theta$,*

- if $\Delta \Vdash^{\text{kv}} \kappa$, then $\Theta \Vdash^{\text{kv}} \kappa$;

- if $\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \kappa$, then $\Theta \Vdash^{\text{kc}} \sigma \Leftarrow \kappa$.

Definition D.11 (Contextual Size).

$$\begin{aligned} |\Delta \vdash \star| &= 1 \\ |\Delta \vdash \kappa_1 \rightarrow \kappa_2| &= 1 + |\Delta \vdash \kappa_1| + |\Delta \vdash \kappa_2| \\ |\Delta[\widehat{\alpha}] \vdash \widehat{\alpha}| &= 1 \\ |\Delta[\widehat{\alpha} = \kappa] \vdash \widehat{\alpha}| &= 1 + |\Delta[\widehat{\alpha} = \kappa] \vdash \kappa| \end{aligned}$$

Lemma D.12 (Substitution Kinding). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa$, then $\Delta \Vdash^{\text{kv}} [\Delta]\kappa$.*

Lemma D.13 (Context Extension with Defaulting is Context Extension). *If $\Delta \longrightarrow \Theta$, then $\Delta \longrightarrow \Theta$.*

Lemma D.14 (Reflexivity of Context Extension). *If Δ ok, then $\Delta \longrightarrow \Delta$.*

Lemma D.15 (Well-formedness of Context Extension). *If Δ ok, and $\Delta \longrightarrow \Theta$, then Θ ok.*

Definition D.16 (Softness). *A context Δ is soft iff it contains only of $\widehat{\alpha}$ and $\widehat{\alpha} = \kappa$ declarations.*

Lemma D.17 (Extension Order).

- (1) *If $\Delta_1, a : \kappa, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, a : \kappa, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 soft, then Θ_2 soft.*
- (2) *If $\Delta_1, T : \kappa, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, T : \kappa, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 soft, then Θ_2 soft.*
- (3) *If $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \Theta', \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$, and Θ' is either $\widehat{\alpha}$ or $\widehat{\alpha} = \kappa$ for some κ . Moreover, if Δ_2 soft, then Θ_2 soft.*
- (4) *If $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$, and $[\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$. Moreover, if Δ_2 soft, then Θ_2 soft.*

Lemma D.18 (Substitution Extension Invariance). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa$, and $\Delta \longrightarrow \Theta$, then $[\Theta]\kappa = [\Theta]([\Delta]\kappa)$ and $[\Theta]\kappa = [\Delta]([\Theta]\kappa)$. As a corollary, if $\Delta \Vdash^{\text{kv}} \kappa_1$, $\Delta \Vdash^{\text{kv}} \kappa_2$, and $[\Delta]\kappa_1 = [\Delta]\kappa_2$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2$.*

Lemma D.19 (Substitution Stability). *If Δ_1, Δ_2 ok, and $\Delta_1 \Vdash^{\text{kv}} \kappa$, then $[\Delta_1, \Delta_2]\kappa = [\Delta_1]\kappa$.*

Lemma D.20 (Transitivity of Context Extension). *If Δ' ok, and $\Delta' \longrightarrow \Delta$, and $\Delta \longrightarrow \Theta$, then $\Delta' \longrightarrow \Theta$.*

Lemma D.21 (Solution Admissibility for Extension). *If $\Delta_1, \widehat{\alpha}, \Delta_2$ ok and $\Delta_1 \Vdash^{\text{kv}} \kappa$, then $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} = \kappa, \Delta_2$.*

Lemma D.22 (Solved Variable Addition for Extension). *If Δ_1, Δ_2 ok and $\Delta_1 \Vdash^{\text{kv}} \kappa$, then $\Delta_1, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} = \kappa, \Delta_2$.*

Lemma D.23 (Unsolved Variable Addition). *If Δ_1, Δ_2 ok then $\Delta_1, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha}, \Delta_2$.*

Lemma D.24 (Parallel Admissibility). *If $\Delta_1 \longrightarrow \Theta_1$, and Δ_1, Δ_2 ok, and $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$, and Δ_2 is fresh w.r.t. Θ_1 , then:*

- $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha}, \Theta_2$

- If $\Theta_1 \Vdash^{\text{kv}} \kappa$, then $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa, \Theta_2$
- If $[\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$, then $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$

Lemma D.25 (Parallel Extension Solution). *If $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$, and $[\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$, then $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$.*

Lemma D.26 (Parallel Variable Update). *If $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa, \Theta_2$, and $\Delta_1 \Vdash^{\text{kv}} \kappa_1$, and $\Theta_1 \Vdash^{\text{kv}} \kappa_2$, and $[\Theta_1]\kappa = [\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$, then $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$.*

D.1.4 Properties of Complete Context.

Lemma D.27 (Type Constructor Preservation). *If Δ ok, and $(T : \kappa) \in \Delta$, and $\Delta \longrightarrow \Omega$, then $(T : [\Omega]\kappa) \in [\Omega]\Delta$.*

Lemma D.28 (Type Variable Preservation). *If Δ ok, and $(a : \kappa) \in \Delta$, and $\Delta \longrightarrow \Omega$, then $(a : [\Omega]\kappa) \in [\Omega]\Delta$.*

Lemma D.29 (Finishing Kinding). *If Ω ok, and $\Omega \Vdash^{\text{kv}} \kappa$, and $\Omega \longrightarrow \Omega'$, then $[\Omega]\kappa = [\Omega']\kappa$.*

Lemma D.30 (Finishing Term Contexts). *If Ω ok, and $\Omega \Vdash^{\text{ectx}} \Gamma$, and $\Omega \longrightarrow \Omega'$, then $[\Omega']\Gamma = [\Omega]\Gamma$.*

Lemma D.31 (Stability of Complete Contexts). *If $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta = [\Omega]\Omega$.*

Lemma D.32 (Softness Goes Away). *If $\Delta_1, \Delta_2 \longrightarrow \Omega_1, \Omega_2$ where $\Delta_1 \longrightarrow \Omega_1$, and Δ_2 soft, then $[\Omega_1, \Omega_2](\Delta_1, \Delta_2) = [\Omega_1]\Delta_1$.*

Lemma D.33 (Confluence of Completeness). *If $\Delta_1 \longrightarrow \Omega$, and $\Delta_2 \longrightarrow \Omega$, then $[\Omega]\Delta_1 = [\Omega]\Delta_2$.*

Lemma D.34 (Finishing Completions). *If Ω ok, and $\Omega \longrightarrow \Omega'$, then $[\Omega]\Omega = [\Omega']\Omega'$.*

D.1.5 Soundness of Algorithm.

Lemma D.35 (Soundness of Kind Validating). *If Ω ok, and $\Omega \Vdash^{\text{kv}} \kappa$, then $[\Omega]\kappa$ is a validate kind in the declarative system.*

Lemma D.36 (Soundness of Well-formed Type Context). *If Δ ok, and $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta$ is a valid type context in the declarative system.*

Lemma D.37 (Soundness of Well-formed Term Context). *If Δ ok, and $\Delta \Vdash^{\text{ectx}} \Gamma$, and $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta \vdash [\Omega]\Gamma$.*

Lemma D.38 (Soundness of Promotion). *If Δ ok, and $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa_1 \rightsquigarrow \kappa_2 \dashv \Theta$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2$. Moreover, if $\Delta \Vdash^{\text{kv}} \kappa_1$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\kappa_1 = [\Omega]\kappa_2$.*

Lemma D.39 (Soundness of Unification). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa_1$, and $\Delta \Vdash^{\text{kv}} \kappa_2$, and $\Delta \Vdash^{\mu} \kappa_1 \approx \kappa_2 \dashv \Theta$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\kappa_1 = [\Omega]\kappa_2$.*

Lemma D.40 (Soundness of Application Kinding). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa_1$, and $\Delta \Vdash^{\text{kv}} \kappa_2$, and $\Delta \Vdash^{\text{kapp}} \kappa_1 \bullet \kappa_2 : \kappa_3 \dashv \Theta$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2 \rightarrow [\Theta]\kappa_3$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\kappa_1 = [\Omega]\kappa_2 \rightarrow [\Omega]\kappa_3$.*

Lemma D.41 (Soundness of Kinding). *If Δ ok, and $\Delta \Vdash^k \sigma : \kappa \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \vdash^k [\Omega]\sigma : [\Omega]\kappa$.*

Lemma D.42 (Soundness of Typing Data Constructor Declaration). *If Δ ok, and $\Delta \Vdash_{\tau'}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash_{\tau'}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau$.*

Lemma D.43 (Soundness of Typing Datatype Declaration). *If Δ ok, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow [\Omega]\Gamma$.*

Lemma D.44 (Soundness of Typing Program). *If Ω ok, and $\Omega \Vdash^{\text{ectx}} \Gamma$, and $\Omega; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$, then $[\Omega]\Omega; [\Omega]\Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$.*

D.1.6 Completeness of Algorithm.

Lemma D.45 (Completeness of Promotion). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{kv}} \widehat{\alpha}$, and $\Delta \Vdash^{\text{kv}} \kappa$, and $[\Delta]\widehat{\alpha} = \widehat{\alpha}$, and $[\Delta]\kappa = \kappa$, if κ is free of $\widehat{\alpha}$, then there exists κ_2, Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$, and $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa \rightsquigarrow \kappa_2 \dashv \Theta$.*

Lemma D.46 (Completeness of Unification). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{kv}} \kappa_1$ and $\Delta \Vdash^{\text{kv}} \kappa_2$, and $[\Delta]\kappa_1 = \kappa_1$ and $[\Delta]\kappa_2 = \kappa_2$, if $[\Omega]\kappa_1 = [\Omega]\kappa_2$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\mu} \kappa_1 \approx \kappa_2 \dashv \Theta$.*

Lemma D.47 (Completeness of Application Kinding). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{kv}} \kappa$ and $\Delta \Vdash^{\text{kv}} \kappa'$, and $[\Delta]\kappa = \kappa$ and $[\Delta]\kappa' = \kappa'$, if $[\Omega]\kappa = [\Omega]\kappa' \rightarrow \kappa_1$, then there exists κ_2, Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\text{kapp}} \kappa \bullet \kappa' : \kappa_2 \dashv \Theta$, and $[\Omega']\kappa_2 = \kappa_1$.*

Lemma D.48 (Completeness of Kinding). *Given Δ ok and $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \vdash^k [\Omega]\sigma : \kappa$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^k \sigma : \kappa' \dashv \Theta$, and $[\Omega']\kappa' = \kappa$.*

Lemma D.49 (Completeness of Typing Data Constructor Declaration). *Given Δ ok and $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash_{\tau'}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash_{\tau'}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau \dashv \Theta$.*

Lemma D.50 (Completeness of Typing Datatype Declaration). *Given Δ ok, and $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$ and $\Psi = [\Omega']\Gamma$.*

Theorem D.51 (Completeness of Typing a Group). *Given Ω ok, if $[\Omega]\Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i; \overline{\Psi}_i^i$, then there exists $\overline{\kappa}'_i, \overline{\Gamma}_i, \Theta$, and Ω' , such that $\Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}'_i; \overline{\Gamma}_i \dashv \Theta$, where $\Theta \longrightarrow \Omega'$, and $[\Omega']\overline{\kappa}'_i = \overline{\kappa}_i^i$, and $\overline{\Psi}_i = [\Omega']\overline{\Gamma}_i$.*

D.2 Proofs

D.2.1 Well-formedness of Declarative Type System.

Lemma D.1 (Well-formedness of Declarative Typing Data Constructor Declaration). *If $\Sigma \Vdash_{\tau_1}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau_2$, then $\Sigma \vdash^k \tau_2 : \star$.*

PROOF. We have

$$\frac{\text{DC-DECL} \quad \Sigma \vdash^k \overline{\tau}_i^i \rightarrow \tau : \star}{\Sigma \vdash_{\tau}^{\text{dc}} D \overline{\tau}_i^i \rightsquigarrow \overline{\tau}_i^i \rightarrow \tau}$$

The goal follows trivially. \square

Lemma D.2 (Well-formedness of Declarative Typing Datatype Declaration). *If $\Sigma \vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, then $\Sigma \vdash \Psi$.*

PROOF. We have

$$\frac{\text{DT-DECL} \quad (T : \overline{\kappa}_i^i \rightarrow \star) \in \Sigma \quad \overline{\Sigma, \overline{a}_i : \overline{\kappa}_i^i \vdash_{T \overline{a}_i}^{\text{dc}} \mathcal{D}_j \rightsquigarrow \tau_j}}{\Sigma \vdash^{\text{dt}} \mathbf{data} T \overline{a}_i^i = \overline{\mathcal{D}_j^j} \rightsquigarrow D_j : \forall \overline{a}_i : \overline{\kappa}_i^i. \tau_j}$$

$\overline{\Sigma, \overline{a}_i : \overline{\kappa}_i^i \vdash^k \tau_j : \star}$	By Lemma D.1
$\overline{\Sigma \vdash^k \forall \overline{a}_i : \overline{\kappa}_i^i. \tau_j : \star}$	By rule K-FORALL
$\overline{\Sigma \vdash D_j : \forall \overline{a}_i : \overline{\kappa}_i^i. \tau_j}$	By rule ECTX-DCON

\square

D.2.2 Well-formedness of Algorithmic Type System. By Lemma D.15 we know that if Δ ok, and $\Delta \longrightarrow \Theta$, it follows that Θ ok. Therefore, in the following lemma when we have Δ ok and $\Delta \longrightarrow \Theta$, we always implicitly derive that Θ ok.

Lemma D.3 (Well-formedness of Promotion). *If $\Delta_1, \widehat{\alpha}, \Delta_2$ ok, and $\Delta_1, \widehat{\alpha}, \Delta_2 \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa_1 \rightsquigarrow \kappa_2 \dashv \Theta$, then $\Theta = \Theta_1, \widehat{\alpha}, \Theta_2$, and $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta$, and $\Theta_1 \Vdash^{\text{kv}} \kappa_2$, and Θ ok. By weakening, there is also $\Theta \Vdash^{\text{kv}} \kappa_2$.*

PROOF. By induction on promotion.

- Case

$$\frac{\text{A-PR-STAR}}{\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \star \rightsquigarrow \star \dashv \Delta}$$

The goals hold trivially.

- Case

$$\frac{\text{A-PR-ARROW} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa_1 \rightsquigarrow \kappa_3 \dashv \Delta_1 \quad \Delta_1 \Vdash_{\widehat{\alpha}}^{\text{pr}} [\Delta_1] \kappa_2 \rightsquigarrow \kappa_4 \dashv \Theta}{\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa_1 \rightarrow \kappa_2 \rightsquigarrow \kappa_3 \rightarrow \kappa_4 \dashv \Theta}$$

$\Delta \longrightarrow \Delta_1 \wedge \Delta_1 = \Delta_{11}, \widehat{\alpha}, \Delta_{12} \wedge \Delta_{11} \Vdash^{\text{kv}} \kappa_3$	I.H.
$\Delta_1 \longrightarrow \Theta \wedge \Theta = \Theta_1, \widehat{\alpha}, \Theta_2 \wedge \Theta_1 \Vdash^{\text{kv}} \kappa_4$	I.H.
$\Delta \longrightarrow \Theta$	By Lemma D.20
$\Delta_{11} \longrightarrow \Theta_1$	By Lemma D.17
$\Theta_1 \Vdash^{\text{kv}} \kappa_3$	By Lemma D.10
$\Theta_1 \Vdash^{\text{kv}} \kappa_3 \rightarrow \kappa_4$	By rule A-KV-ARROW

- Case

$$\frac{\text{A-PR-KUVARL}}{\Delta[\widehat{\beta}][\widehat{\alpha}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \widehat{\beta} \rightsquigarrow \widehat{\beta} \dashv \Delta[\widehat{\beta}][\widehat{\alpha}]}$$

The goals hold trivially.

- Case

$$\frac{\text{A-PR-KUVARR}}{\Delta[\widehat{\alpha}][\widehat{\beta}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_1 \dashv \Delta[\widehat{\beta}_1, \widehat{\alpha}][\widehat{\beta} = \widehat{\beta}_1]}$$

Most goals hold trivially. By Lemmas D.21 and D.23 and transitivity (Lemma D.20) we can prove $\Delta[\widehat{\alpha}][\widehat{\beta}] \longrightarrow \Delta[\widehat{\beta}_1, \widehat{\alpha}][\widehat{\beta} = \widehat{\beta}_1]$.

□

Lemma D.4 (Well-formedness of Unification). *If Δ ok, and $\Delta \Vdash^{\mu} \kappa_1 \approx \kappa_2 \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok.*

PROOF. By induction on the derivation of kind unification.

- Case

$$\frac{\text{A-U-REFL}}{\Delta \Vdash^{\mu} \kappa \approx \kappa \dashv \Delta}$$

$\Delta \longrightarrow \Delta$ | By Lemma D.14

- Case

$$\frac{\text{A-U-ARROW} \quad \Delta \Vdash^{\mu} \kappa_1 \approx \kappa_3 \dashv \Theta_1 \quad \Theta_1 \Vdash^{\mu} [\Theta_1] \kappa_2 \approx [\Theta_1] \kappa_4 \dashv \Theta}{\Delta \Vdash^{\mu} \kappa_1 \rightarrow \kappa_2 \approx \kappa_3 \rightarrow \kappa_4 \dashv \Theta}$$

$\Delta \longrightarrow \Theta_1$ | By I.H.
 $\Theta_1 \longrightarrow \Theta$ | By I.H.
 $\Delta \longrightarrow \Theta$ | By Lemma D.20

- Case

$$\frac{\text{A-U-KVARL} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa \rightsquigarrow \kappa_2 \dashv \Theta[\widehat{\alpha}]}{\Delta[\widehat{\alpha}] \Vdash^{\mu} \widehat{\alpha} \approx \kappa \dashv \Theta[\widehat{\alpha} = \kappa_2]}$$

$\Theta = \Theta_1, \widehat{\alpha}, \Theta_2 \wedge \Delta \longrightarrow \Theta[\widehat{\alpha}] \wedge \Theta_1 \Vdash^{\text{kv}} \kappa_2$ | By Lemma D.3
 $\Theta \longrightarrow \Theta[\widehat{\alpha} = \kappa_2]$ | By Lemma D.21
 $\Delta \longrightarrow \Theta[\widehat{\alpha} = \kappa_2]$ | By Lemma D.20

- Case

$$\frac{\text{A-U-KVARR} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \kappa \rightsquigarrow \kappa_2 \dashv \Theta[\widehat{\alpha}]}{\Delta[\widehat{\alpha}] \Vdash^{\mu} \kappa \approx \widehat{\alpha} \dashv \Theta[\widehat{\alpha} = \kappa_2]}$$

Similar to the previous case.

□

Lemma D.5 (Well-formedness of Application Kinding). *If Δ ok, and $\Delta \Vdash^{\text{kapp}} \kappa_1 \bullet \kappa_2 : \kappa \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok. Moreover, if $\Delta \Vdash^{\text{kv}} \kappa_1$, then we have $\Theta \Vdash^{\text{kv}} \kappa$.*

PROOF. By induction on the derivation of application kinding.

- Case

$$\frac{\text{A-KAPP-KUVAR} \quad \Delta[\widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha} = \widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2] \Vdash^{\mu} \widehat{\alpha}_1 \approx \kappa \dashv \Theta}{\Delta[\widehat{\alpha}] \Vdash^{\text{kapp}} \widehat{\alpha} \bullet \kappa : \widehat{\alpha}_2 \dashv \Theta}$$

$$\begin{array}{l|l} \Delta[\widehat{\alpha}] \longrightarrow \Delta[\widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha}] & \text{By Lemma D.23} \\ \Delta[\widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha}] \longrightarrow \Delta[\widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha} = \widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2] & \text{By Lemma D.21} \\ \Delta[\widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha} = \widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2] \longrightarrow \Theta & \text{By Lemma D.4} \\ \Delta \longrightarrow \Theta & \text{By Lemma D.20} \\ \Delta[\widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha} = \widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2] \Vdash^{\text{kv}} \widehat{\alpha}_2 & \text{By rule A-KV-KUVAR} \\ \Theta \Vdash^{\text{kv}} \widehat{\alpha}_2 & \text{By Lemma D.10} \end{array}$$

- Case

$$\frac{\text{A-KAPP-ARROW} \quad \Delta \Vdash^{\mu} \kappa_1 \approx \kappa \dashv \Theta}{\Delta \Vdash^{\text{kapp}} \kappa_1 \rightarrow \kappa_2 \bullet \kappa : \kappa_2 \dashv \Theta}$$

$$\begin{array}{l|l} \Delta \longrightarrow \Theta & \text{By Lemma D.4} \\ \Delta \Vdash^{\text{kv}} \kappa_1 \rightarrow \kappa_2 & \text{Given} \\ \Delta \Vdash^{\text{kv}} \kappa_2 & \text{By inversion} \\ \Theta \Vdash^{\text{kv}} \kappa_2 & \text{By Lemma D.10} \end{array}$$

□

Lemma D.6 (Well-formedness of Kinding). *If Δ ok, and $\Delta \Vdash^{\text{k}} \sigma : \kappa \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok, and $\Theta \Vdash^{\text{kv}} \kappa$ and $\Theta \Vdash^{\text{k}} \sigma \Leftarrow \kappa$.*

PROOF. By induction on the derivation of kinding.

- Case for rules **A-K-NAT**, **A-K-VAR**, **A-K-TCON**, and **A-K-ARROW** follows trivially.
- Case

$$\frac{\text{A-K-FORALL} \quad \Delta \Vdash^{\text{kv}} \kappa \quad \Delta, a : \kappa \Vdash^{\text{k}} \sigma : \kappa_2 \dashv \Theta, a : \kappa \quad [\Theta] \kappa_2 = \star}{\Delta \Vdash^{\text{k}} \forall a : \kappa. \sigma : \star \dashv \Theta}$$

$$\begin{array}{l|l} \Delta, a : \kappa \longrightarrow \Theta, a : \kappa \wedge \Theta, a : \kappa \Vdash^{\text{k}} \sigma \Leftarrow \star & \text{By I.H.} \\ \Delta \longrightarrow \Theta & \text{By inversion} \\ \Theta \Vdash^{\text{kv}} \star & \text{By rule A-KV-STAR} \\ \Delta \Vdash^{\text{kv}} \kappa & \text{Given} \\ \Theta \Vdash^{\text{kv}} \kappa & \text{By Lemma D.10} \\ \Theta \Vdash^{\text{k}} \forall a : \kappa. \sigma \Leftarrow \star & \text{By rules A-KC-EQ and A-K-FORALL} \end{array}$$

- Case

$$\frac{\text{A-K-APP} \quad \Delta \Vdash^{\text{k}} \tau_1 : \kappa_1 \dashv \Theta_1 \quad \Theta_1 \Vdash^{\text{k}} \tau_2 : \kappa_2 \dashv \Theta_2 \quad \Theta_2 \Vdash^{\text{kapp}} [\Theta_2] \kappa_1 \bullet [\Theta_2] \kappa_2 : \kappa_3 \dashv \Theta}{\Delta \Vdash^{\text{k}} \tau_1 \tau_2 : \kappa_3 \dashv \Theta}$$

$$\begin{array}{l|l} \Delta \longrightarrow \Theta_1 \wedge \Theta_2 \Vdash^{\text{kv}} \kappa_1 \wedge \Theta_1 \Vdash^{\text{k}} \tau_1 \Leftarrow \kappa_1 & \text{I.H.} \\ \Theta_1 \longrightarrow \Theta_2 \wedge \Theta_2 \Vdash^{\text{kv}} \kappa_2 \wedge \Theta_2 \Vdash^{\text{k}} \tau_2 \Leftarrow \kappa_2 & \text{I.H.} \\ \Theta_2 \longrightarrow \Theta & \text{By Lemma D.5} \end{array}$$

$\Theta_1 \longrightarrow \Theta$	By Lemma D.20
$\Delta \longrightarrow \Theta$	By Lemma D.20
$\Theta_2 \Vdash^{\text{kv}} \kappa_1$	By Lemma D.10
$\Theta_2 \Vdash^{\text{kv}} [\Theta_2]\kappa_1$	By Lemma D.12
$\Theta \Vdash^{\text{kv}} \kappa_3$	By Lemma D.5
$\Theta \Vdash^{\text{kc}} \tau_1 \Leftarrow \kappa_1$	By Lemma D.10
$\Theta \Vdash^{\text{kc}} \tau_2 \Leftarrow \kappa_2$	By Lemma D.10
$[\Theta]([\Theta_2]\kappa_1) = [\Theta]([\Theta_2]\kappa_2) \rightarrow [\Theta]\kappa_3$	By Lemma D.40
$[\Theta]\kappa_1 = [\Theta]\kappa_2 \rightarrow [\Theta]\kappa_3$	By Lemma D.18
$\Theta \Vdash^{\text{kapp}} [\Theta]\kappa_1 \bullet [\Theta]\kappa_2 : [\Theta]\kappa_3 \dashv \Theta$	By rule A-KAPP-ARROW and rule A-U-REFL
$\Theta \Vdash^{\text{kc}} \tau_1 \tau_2 \Leftarrow \kappa_3$	By rules A-KC-EQ and A-K-APP

□

Lemma D.7 (Well-formedness of Typing Data Constructor Declarations). *If Δ ok, and $\Delta \Vdash_{\tau}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok. and $\Theta \Vdash^{\text{kc}} \tau \Leftarrow \star$.*

PROOF.

$$\frac{\text{A-DC-DECL} \quad \Delta \Vdash^{\text{k}} \overline{\tau}_i^i \rightarrow \tau : \star \dashv \Theta}{\Delta \Vdash_{\tau}^{\text{dc}} D \overline{\tau}_i^i \rightsquigarrow \overline{\tau}_i^i \rightarrow \tau \dashv \Theta}$$

Follows directly from Lemma D.6.

□

Lemma D.8 (Well-formedness of Typing Datatype Declaration). *If Δ ok, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok, and $\Theta \Vdash^{\text{ectx}} \Gamma$.*

PROOF.

A-DT-DECL

$$\frac{(T : \kappa) \in \Delta \quad \Delta, \overline{\widehat{a}}_i^i \Vdash^{\text{u}} [\Delta]\kappa \approx (\overline{\widehat{a}}_i^i \rightarrow \star) \dashv \Theta_1, \overline{\widehat{a}}_i^i = \kappa_i^i \quad \overline{\Theta_j, \overline{a}_i : \kappa_i^i \Vdash_{T \overline{a}_i^i}^{\text{dc}} \mathcal{D}_j \rightsquigarrow \tau_j \dashv \Theta_{j+1}, \overline{a}_i : \kappa_i^i}^j}{\Delta \Vdash^{\text{dt}} \text{data } T \overline{a}_i^i = \overline{\mathcal{D}_j}^{j \in 1..n} \rightsquigarrow \overline{D_j : \forall \overline{a}_i : \kappa_i^i . \tau_j}^j \dashv \Theta_{n+1}}$$

$\Delta \longrightarrow \Delta, \overline{\widehat{a}}_i^i$	By rule A-CTXE-ADD
$\Delta, \overline{\widehat{a}}_i^i \longrightarrow \Theta_1, \overline{\widehat{a}}_i^i = \kappa_i^i$	By Lemma D.4
$\Delta \longrightarrow \Theta_1$	By Lemma D.17
$\Theta_1, \overline{a}_i : \kappa_i^i \longrightarrow \Theta_{n+1}, \overline{a}_i : \kappa_i^i$	By Lemma D.7 and Lemma D.20
$\Theta_1 \longrightarrow \Theta_{n+1}$	By inversion
$\Delta \longrightarrow \Theta$	By Lemma D.20
$\overline{\Theta_{j+1}, \overline{a}_i : \kappa_i^i \Vdash^{\text{kc}} \tau_j \Leftarrow \star}^{j \in 1..n}$	By Lemma D.7
$\overline{\Theta_n, \overline{a}_i : \kappa_i^i \Vdash^{\text{kc}} \tau_j \Leftarrow \star}^{j \in 1..n}$	By Lemma D.10
$\overline{\Theta_n \Vdash^{\text{kc}} \forall \overline{a}_i : \kappa_i^i . \tau_j \Leftarrow \star}^{j \in 1..n}$	By rules A-KC-EQ and A-K-FORALL
$\Theta_n \Vdash^{\text{ectx}} D_j : \forall \overline{a}_i : \overline{\widehat{a}}_i^i . \tau_j$	By rule A-ECTX-DCON

□

D.2.3 Properties of Context Extension.

Lemma D.9 (Declaration Preservation). *If $\Delta \longrightarrow \Theta$, if a type constructor or a type variable or a kind unification variable is declared in Δ , then it is declared in Θ .*

PROOF. By a straightforward induction on $\Delta \longrightarrow \Theta$. □

Lemma D.10 (Extension Weakening). *Given $\Delta \longrightarrow \Theta$,*

- *if $\Delta \Vdash^{\text{kv}} \kappa$, then $\Theta \Vdash^{\text{kv}} \kappa$;*
- *if $\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \kappa$, then $\Theta \Vdash^{\text{kc}} \sigma \Leftarrow \kappa$.*

PROOF. **Part 1** By induction on $\Delta \Vdash^{\text{kv}} \kappa$.

- Case

$$\frac{\text{A-KV-STAR}}{\Delta \Vdash^{\text{kv}} \star}$$

The goal holds trivially.

- Case

$$\frac{\text{A-KV-ARROW} \quad \Delta \Vdash^{\text{kv}} \kappa_1 \quad \Delta \Vdash^{\text{kv}} \kappa_2}{\Delta \Vdash^{\text{kv}} \kappa_1 \rightarrow \kappa_2}$$

The goal holds directly from I.H..

- Case

$$\frac{\text{A-KV-KUVAR} \quad \widehat{\alpha} \in \Delta}{\Delta \Vdash^{\text{kv}} \widehat{\alpha}}$$

The goal holds directly from Lemma D.9.

Part 2 By induction on $\Delta \Vdash^{\text{k}} \sigma : \kappa \dashv \Delta$.

- The case for rules **A-K-NAT** and **A-K-ARROW** holds trivially.
- The case for rules **A-K-VAR** and **A-K-TCON** holds from Lemma D.7 and Lemma D.18.
- The case for rule **A-K-FORALL** holds from I.H. and Lemma D.18.
- The case for rule **A-K-APP** depends on the extension weakening of application kinding. Given the hypothesis, it's impossible for the derivation to ever use rule **A-KAPP-KUVAR**. The extension weakening on rule **A-KAPP-ARROW** then depends on the extension weakening of kind unification. Given the hypothesis, it's impossible for the derivation to ever use rules **A-U-KVARL** and **A-U-KVARR**. The case for rule **A-U-RREFL** holds trivially, and the case for rule **A-U-ARROW** holds directly from I.H. □

Lemma D.12 (Substitution Kinding). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa$, then $\Delta \Vdash^{\text{kv}} [\Delta]\kappa$.*

PROOF. By induction on $|\Delta \vdash \kappa|$. We then case analyze κ .

- $\kappa = \star$. The goal holds trivially.
- $\kappa = \kappa_1 \rightarrow \kappa_2$. The goal directly from I.H..
- $\kappa = \widehat{\alpha}$. If $\widehat{\alpha}$ is unsolved in Δ , then the goal holds directly. Or otherwise we have $\Delta = \Delta_1, \widehat{\alpha} = \kappa, \Delta_2$. Because Δ ok, we have $\Delta_1 \Vdash^{\text{kv}} \kappa$ and $|\Delta_1 \vdash \kappa| = |\Delta \vdash \kappa|$, which is less than $|\Delta \vdash \widehat{\alpha}|$. So we apply I.H. to get the goal. □

Lemma D.13 (Context Extension with Defaulting is Context Extension). *If $\Delta \longrightarrow \Theta$, then $\Delta \longrightarrow \Theta$.*

PROOF. By straightforward induction on $\Delta \longrightarrow \Theta$. □

Lemma D.14 (Reflexivity of Context Extension). *If Δ ok, then $\Delta \longrightarrow \Delta$.*

PROOF. By straightforward induction on Δ ok. The conclusion follows directly from the definition. \square

Lemma D.15 (Well-formedness of Context Extension). *If Δ ok, and $\Delta \longrightarrow \Theta$, then Θ ok.*

PROOF. By induction on $\Delta \longrightarrow \Theta$.

- Case

$$\frac{\text{A-CTXE-EMPTY}}{\bullet \longrightarrow \bullet}$$

Follows directly by rule **A-TCTX-EMPTY**.

- Case

$$\frac{\text{A-CTXE-TVAR} \quad \Delta \longrightarrow \Theta}{\Delta, a : \kappa \longrightarrow \Theta, a : \kappa}$$

$\Delta, a : \kappa$ ok	Given
$\Delta \Vdash^{\text{kv}} \kappa$	By inversion
$\Delta \longrightarrow \Theta$	Given
$\Theta \Vdash^{\text{kv}} \kappa$	By lemma D.10
Θ ok	I.H.
$\Theta, a : \kappa$ ok	By rule A-TCTX-TVAR

- Case

$$\frac{\text{A-CTXE-TCON} \quad \Delta \longrightarrow \Theta}{\Delta, T : \kappa \longrightarrow \Theta, T : \kappa}$$

This case is similar to the case for rule **A-TCTX-TVAR**.

- Case

$$\frac{\text{A-CTXE-KUVAR} \quad \Delta \longrightarrow \Theta}{\Delta, \widehat{a} \longrightarrow \Theta, \widehat{a}}$$

The goal holds directly from I.H. and rule **A-TCTX-KUVAR**.

- Case

$$\frac{\text{A-CTXE-KUVAR}^{\text{SOLVED}} \quad \Delta \longrightarrow \Theta \quad [\Theta]\kappa_1 = [\Theta]\kappa_2}{\Delta, \widehat{a} = \kappa_1 \longrightarrow \Theta, \widehat{a} = \kappa_2}$$

Θ ok	I.H.
$\Delta, \widehat{a} = \kappa_1$ ok	Given
$\Delta \Vdash^{\text{kv}} \kappa_1$	By inversion
$\Delta \longrightarrow \Theta$	Given
$\Theta \Vdash^{\text{kv}} \kappa_1$	By lemma D.10

Suppose κ_2 is not well-formed under Θ , then it must contain kind unification variables that are not in Θ . Then it is impossible to have $[\Theta]\kappa_1 = [\Theta]\kappa_2$ given $\Theta \Vdash^{\text{kv}} \kappa_1$. Thus by contradiction we have $\Theta \Vdash^{\text{kv}} \kappa_2$. Then $\Theta, \widehat{a} = \kappa_2$ ok by rule **A-TCTX-TCON**.

- Case

$$\frac{\text{A-CTXE-SOLVE} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha} = \kappa}$$

The goal holds directly from I.H. and rule **A-TCTX-KUVAR****SOLVED**.

- Case

$$\frac{\text{A-CTXE-ADD} \quad \Delta \longrightarrow \Theta}{\Delta \longrightarrow \Theta, \widehat{\alpha}}$$

The goal holds directly from I.H. and rule **A-TCTX-KUVAR**.

- Case

$$\frac{\text{A-CTXE-ADDSOLVED} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta \longrightarrow \Theta, \widehat{\alpha} = \kappa}$$

The goal holds directly from I.H. and rule **A-TCTX-KUVAR****SOLVED**.

□

Lemma D.17 (Extension Order).

- (1) If $\Delta_1, a : \kappa, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, a : \kappa, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.
- (2) If $\Delta_1, T : \kappa, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, T : \kappa, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.
- (3) If $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \Theta', \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$, and Θ' is either $\widehat{\alpha}$ or $\widehat{\alpha} = \kappa$ for some κ . Moreover, if Δ_2 **soft**, then Θ_2 **soft**.
- (4) If $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$, and $[\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.

PROOF. We give the detailed proof for the first part. The proof for the rest parts is similar.

By induction on $\Delta_1, a : \kappa, \Delta_2 \longrightarrow \Theta$.

- Case $\Delta = \bullet$ by rule **A-CTXE-EMPTY**. This case is impossible.
- Case $\Delta_1, a : \kappa \longrightarrow \Theta', a : \kappa$ by rule **A-CTXE-TVAR** when Δ_2 is empty. In this case, let $\Theta_1 = \Theta'$ and Θ_2 be empty. All goals follow directly.
- Case $\Delta_1, a : \kappa, \Delta', b : \kappa_2 \longrightarrow \Theta', b : \kappa_2$ by rule **A-CTXE-TVAR** where $\Delta_2 = \Delta', b : \kappa_2$ and $\Delta_1, a : \kappa, \Delta' \longrightarrow \Theta'$. By I.H. we have $\Theta' = \Theta_1, a : \kappa, \Theta'_2$ and $\Delta_1 \longrightarrow \Theta_1$. Let $\Theta_2 = \Theta'_2, b : \kappa_2$ and all goals follow directly.
- Case $\Delta_1, a : \kappa, \Delta', T : \kappa_2 \longrightarrow \Theta', T : \kappa_2$ by rule **A-CTXE-TCON** where $\Delta_2 = \Delta', T : \kappa_2$ and $\Delta_1, a : \kappa, \Delta' \longrightarrow \Theta'$. This case is similar to the above case.
- Case $\Delta_1, a : \kappa, \Delta', \widehat{\alpha} \longrightarrow \Theta', \widehat{\alpha}$ by rule **A-CTXE-KUVAR** where $\Delta_2 = \Delta', \widehat{\alpha}$ and $\Delta_1, a : \kappa, \Delta' \longrightarrow \Theta'$. By I.H. we have $\Theta' = \Theta_1, a : \kappa, \Theta'_2$ and $\Delta_1 \longrightarrow \Theta_1$. Let $\Theta_2 = \Theta'_2, \widehat{\alpha}$ and all goals follow directly. And if Δ' **soft**, by I.H. we have Θ'_2 **soft**. By definition we have Θ_2 **soft**.
- Case for rules **A-CTXE-KUVAR****SOLVED**, **A-CTXE-SOLVE**, **A-CTXE-ADD**, and **A-CTXE-ADDSOLVED** are similar to the above case.

□

Lemma D.18 (Substitution Extension Invariance). If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa$, and $\Delta \longrightarrow \Theta$, then $[\Theta]\kappa = [\Theta]([\Delta]\kappa)$ and $[\Theta]\kappa = [\Delta]([\Theta]\kappa)$. As a corollary, if $\Delta \Vdash^{\text{kv}} \kappa_1$, $\Delta \Vdash^{\text{kv}} \kappa_2$, and $[\Delta]\kappa_1 = [\Delta]\kappa_2$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2$.

PROOF. Because $\Delta \Vdash^{kv} \kappa$, so every solved kind unification variable in Δ is solved in Θ . Therefore $[\Theta]\kappa = [\Delta]([\Theta]\kappa)$.

To show that $[\Theta]\kappa = [\Theta]([\Delta]\kappa)$, we do induction on $|\Delta \vdash \kappa|$.

-

$$\frac{\text{A-KV-STAR}}{\Delta \Vdash^{kv} \star}$$

The goal follows trivially.

-

$$\frac{\text{A-KV-ARROW} \quad \Delta \Vdash^{kv} \kappa_1 \quad \Delta \Vdash^{kv} \kappa_2}{\Delta \Vdash^{kv} \kappa_1 \rightarrow \kappa_2}$$

The goal follows directly from I.H..

-

$$\frac{\text{A-KV-KUVAR} \quad \widehat{\alpha} \in \Delta}{\Delta \Vdash^{kv} \widehat{\alpha}}$$

There are two subcases. Firstly, $\widehat{\alpha}$ is unsolved in Δ . Then $[\Theta]([\Delta]\widehat{\alpha}) = [\Theta]\widehat{\alpha}$ follows directly. Or we have $\Delta = \Delta_1, \widehat{\alpha} = \kappa, \Delta_2$. Then by Lemma D.17 we have $\Theta = \Theta_1, \widehat{\alpha} = \kappa', \Theta_2$ and $[\Theta_1]\kappa = [\Theta_1]\kappa'$. Because $|\Delta \vdash \kappa| < |\Delta \vdash \widehat{\alpha}|$, by I.H., we know that $[\Theta]\kappa = [\Theta]([\Delta]\kappa)$. Therefore, $[\Theta]\widehat{\alpha} = [\Theta]\kappa' = [\Theta]\kappa = [\Theta]([\Delta]\kappa) = [\Theta]([\Delta]\widehat{\alpha})$.

For the corollary, we have $[\Theta]\kappa_1 = [\Theta]([\Delta]\kappa_1) = [\Theta]([\Delta]\kappa_2) = [\Theta]\kappa_2$. □

Lemma D.19 (Substitution Stability). *If Δ_1, Δ_2 ok, and $\Delta_1 \Vdash^{kv} \kappa$, then $[\Delta_1, \Delta_2]\kappa = [\Delta_1]\kappa$.*

PROOF. Follows directly as κ and Δ_1 do not contain kind variables in Δ_2 . □

Lemma D.20 (Transitivity of Context Extension). *If Δ' ok, and $\Delta' \longrightarrow \Delta$, and $\Delta \longrightarrow \Theta$, then $\Delta' \longrightarrow \Theta$.*

PROOF. By induction on $\Delta \longrightarrow \Theta$.

- Case

$$\frac{\text{A-CTXE-EMPTY}}{\bullet \longrightarrow \bullet}$$

We have $\Delta' \longrightarrow \bullet$ as given.

- Case

$$\frac{\text{A-CTXE-TVAR} \quad \Delta \longrightarrow \Theta}{\Delta, a : \kappa \longrightarrow \Theta, a : \kappa}$$

$\Delta' \longrightarrow \Delta, a : \kappa$	Given By inversion I.H. By rule A-CTXE-TVAR
$\Delta' = \Delta_1, a : \kappa \wedge \Delta_1 \longrightarrow \Delta$	
$\Delta_1 \longrightarrow \Theta$	
$\Delta_1, a : \kappa \longrightarrow \Theta, a : \kappa$	

- Case

$$\frac{\text{A-CTXE-TCON} \quad \Delta \longrightarrow \Theta}{\Delta, T : \kappa \longrightarrow \Theta, T : \kappa}$$

This case is similar to the case for rule **A-CTXE-TVAR**.

- Case

$$\frac{\text{A-CTXE-KUVAR} \quad \Delta \longrightarrow \Theta}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha}}$$

Since $\Delta' \longrightarrow \Delta, \widehat{\alpha}$, the derivation must conclude with either rule **A-CTXE-KUVAR** or rule **A-CTXE-ADD**.

– By rule **A-CTXE-KUVAR**.

$$\begin{array}{l|l} \Delta' = \Delta_1, \widehat{\alpha} \wedge \Delta_1 \longrightarrow \Delta & \text{Given} \\ \Delta_1 \longrightarrow \Theta & \text{I.H.} \\ \Delta_1, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha} & \text{By rule A-CTXE-KUVAR} \end{array}$$

– By rule **A-CTXE-ADD**.

$$\begin{array}{l|l} \Delta' \longrightarrow \Delta & \text{Given} \\ \Delta' \longrightarrow \Theta & \text{I.H.} \\ \Delta' \longrightarrow \Theta, \widehat{\alpha} & \text{By rule A-CTXE-ADD} \end{array}$$

- Case

$$\frac{\text{A-CTXE-KUVAR}^{\text{SOLVED}} \quad \Delta \longrightarrow \Theta \quad [\Theta]\kappa_1 = [\Theta]\kappa_2}{\Delta, \widehat{\alpha} = \kappa_1 \longrightarrow \Theta, \widehat{\alpha} = \kappa_2}$$

Since $\Delta' \longrightarrow \Delta, \widehat{\alpha} = \kappa_1$, the derivation must conclude with either rule **A-CTXE-KUVAR**^{SOLVED} or rule **A-CTXE-ADD**^{SOLVED}.

– By rule **A-CTXE-KUVAR**^{SOLVED}.

$$\begin{array}{l|l} \Delta' = \Delta_1, \widehat{\alpha} = \kappa_0 \wedge \Delta_1 \longrightarrow \Delta \wedge [\Delta]\kappa_0 = [\Delta]\kappa_1 & \text{Given} \\ \Delta_1 \longrightarrow \Theta & \text{I.H.} \\ [\Theta]\kappa_0 & \\ = [\Theta]\kappa_1 & \text{By Lemma D.18} \\ = [\Theta]\kappa_2 & \text{Given} \\ \Delta_1, \widehat{\alpha} = \kappa_0 \longrightarrow \Theta, \widehat{\alpha} = \kappa_2 & \text{By rule A-CTXE-KUVAR} \end{array}$$

– By rule **A-CTXE-ADD**^{SOLVED}.

$$\begin{array}{l|l} \Delta' \longrightarrow \Delta & \text{Given} \\ \Delta' \longrightarrow \Theta & \text{I.H.} \\ \Delta' \longrightarrow \Theta, \widehat{\alpha} = \kappa_2 & \text{By rule A-CTXE-ADD}^{\text{SOLVED}} \end{array}$$

- Case

$$\frac{\text{A-CTXE-SOLVE} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha} = \kappa}$$

Since $\Delta' \longrightarrow \Delta, \widehat{\alpha}$, the derivation must conclude with either rule **A-CTXE-KUVAR** or rule **A-CTXE-ADD**.

– By rule **A-CTXE-KUVAR**.

$$\begin{array}{l|l} \Delta' = \Delta_1, \widehat{\alpha} \wedge \Delta_1 \longrightarrow \Delta & \text{Given} \\ \Delta_1 \longrightarrow \Theta & \text{I.H.} \\ \Delta_1, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha} = \kappa & \text{By rule A-CTXE-SOLVE} \end{array}$$

– By rule A-CTXE-ADD.

$$\begin{array}{l|l} \Delta' \longrightarrow \Delta & \text{Given} \\ \Delta' \longrightarrow \Theta & \text{I.H.} \\ \Delta' \longrightarrow \Theta, \widehat{\alpha} = \kappa & \text{By rule A-CTXE-ADDSOLVED} \end{array}$$

• Case

$$\frac{\text{A-CTXE-ADD} \quad \Delta \longrightarrow \Theta}{\Delta \longrightarrow \Theta, \widehat{\alpha}}$$

$$\begin{array}{l|l} \Delta' \longrightarrow \Theta & \text{I.H.} \\ \Delta' \longrightarrow \Theta, \widehat{\alpha} & \text{By rule A-CTXE-ADD} \end{array}$$

• Case

$$\frac{\text{A-CTXE-ADDSOLVED} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta \longrightarrow \Theta, \widehat{\alpha} = \kappa}$$

$$\begin{array}{l|l} \Delta' \longrightarrow \Theta & \text{I.H.} \\ \Delta' \longrightarrow \Theta, \widehat{\alpha} = \kappa & \text{By rule A-CTXE-ADDSOLVED} \end{array}$$

□

Lemma D.21 (Solution Admissibility for Extension). *If $\Delta_1, \widehat{\alpha}, \Delta_2$ ok and $\Delta_1 \Vdash^{\text{kv}} \kappa$, then $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} = \kappa, \Delta_2$.*

PROOF. By induction on Δ_2 .

- Case Δ_2 is empty. Then $\Delta_1 \longrightarrow \Delta_1$ by Lemma D.14, and $\Delta_1, \widehat{\alpha} \longrightarrow \Delta_1, \widehat{\alpha} = \kappa$ holds by rule A-CTXE-SOLVE.
- Case $\Delta_2 = \Delta'_2, a : \kappa$. By I.H., we $\Delta_1, \widehat{\alpha}, \Delta'_2 \longrightarrow \Delta_1, \widehat{\alpha} = \kappa, \Delta'_2$. Then by rule A-CTXE-TVAR we are done.
- Case $\Delta_2 = \Delta'_2, T : \kappa$. By I.H. and rule A-CTXE-TCON.
- Case $\Delta_2 = \Delta'_2, \widehat{\alpha}$. By I.H. and rule A-CTXE-KUVAR.
- Case $\Delta_2 = \Delta'_2, \widehat{\alpha} = \kappa$. By I.H. and rule A-CTXE-KUVAR SOLVED.

□

Lemma D.22 (Solved Variable Addition for Extension). *If Δ_1, Δ_2 ok and $\Delta_1 \Vdash^{\text{kv}} \kappa$, then $\Delta_1, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} = \kappa, \Delta_2$.*

PROOF. The proof is exactly the same as the one for Lemma D.21. Except for the case when Δ_2 is empty, we use rule A-CTXE-ADDSOLVED.

□

Lemma D.23 (Unsolved Variable Addition). *If Δ_1, Δ_2 ok then $\Delta_1, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha}, \Delta_2$.*

PROOF. The proof is exactly the same as the one for Lemma D.21. Except for the case when Δ_2 is empty, we use rule A-CTXE-ADD.

□

Lemma D.24 (Parallel Admissibility). *If $\Delta_1 \longrightarrow \Theta_1$, and Δ_1, Δ_2 ok, and $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$, and Δ_2 is fresh w.r.t. Θ_1 , then:*

- $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha}, \Theta_2$
- If $\Theta_1 \Vdash^{kv} \kappa$, then $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa, \Theta_2$
- If $[\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$, then $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$

PROOF. **Part 1** By induction on Θ_2 .

- $\Theta_2 = \bullet$. Because Δ_2 is fresh w.r.t. Θ_1 , we must have $\Delta_2 = \bullet$. We have $\Delta_1, \widehat{\alpha} \longrightarrow \Theta_1, \widehat{\alpha}$ by rule **A-CTXE-KUVAR**.
- $\Theta_2 = \Theta'_2, a : \kappa$. Then the derivation of $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$ must conclude with rule **A-CTXE-TVAR**. It must be $\Delta_2 = \Delta'_2, a : \kappa$. (Or otherwise if $(a : \kappa) \in \Delta_1$, then we must have $(a : \kappa) \in \Theta_1$ by Lemma D.9, and Θ_1, Θ_2 is no longer well-formed.)

$$\begin{array}{l|l} \Delta_1, \Delta'_2, a : \kappa \longrightarrow \Theta_1, \Theta'_2, a : \kappa & \text{Given} \\ \Delta_1, \Delta'_2 \longrightarrow \Theta_1, \Theta'_2 & \text{By inversion} \\ \Delta_1, \widehat{\alpha}, \Delta'_2 \longrightarrow \Theta_1, \widehat{\alpha}, \Theta'_2 & \text{I.H.} \\ \Delta_1, \widehat{\alpha}, \Delta'_2, a : \kappa \longrightarrow \Theta_1, \widehat{\alpha}, \Theta'_2, a : \kappa & \text{By rule A-CTXE-TVAR} \end{array}$$

- $\Theta_2 = \Theta'_2, T : \kappa$ This case is similar to the case when $\Theta_2 = \Theta'_2, a : \kappa$, except that we reason using rule **A-CTXE-TCON**.
- $\Theta_2 = \Theta'_2, \widehat{\alpha}_1$ Then the derivation of $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$ must conclude with either rule **A-CTXE-KUVAR** or rule **A-CTXE-ADD**.
 - Subcase: the derivation concludes with rule **A-CTXE-KUVAR**. It must be $\Delta_2 = \Delta'_2, \widehat{\alpha}_1$.

$$\begin{array}{l|l} \Delta_1, \Delta'_2 \longrightarrow \Theta_1, \Theta'_2 & \text{Given} \\ \Delta_1, \widehat{\alpha}, \Delta'_2 \longrightarrow \Theta_1, \widehat{\alpha}, \Theta'_2 & \text{I.H.} \\ \Delta_1, \widehat{\alpha}, \Delta'_2, \widehat{\alpha}_1 \longrightarrow \Theta_1, \widehat{\alpha}, \Theta'_2, \widehat{\alpha}_1 & \text{By rule A-CTXE-KUVAR} \end{array}$$

- Subcase: the derivation concludes with rule **A-CTXE-ADD**.

$$\begin{array}{l|l} \Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta'_2 & \text{Given} \\ \Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \Theta'_2 & \text{I.H.} \\ \Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha}, \Theta'_2, \widehat{\alpha}_1 & \text{By rule A-CTXE-ADD} \end{array}$$

- $\Theta_2 = \Theta'_2, \widehat{\alpha}_1 = \kappa$. Then the derivation of $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$ must conclude with either rule **A-CTXE-KUVAR****SOLVED** or rule **A-CTXE-ADD****SOLVED** or rule **A-CTXE-SOLVE**. In either case, the reasoning is similar to the above case.

Part 2 Similar to Part 1, except that when $\Theta_2 = \bullet$, we apply rule **A-CTXE-SOLVE**.

Part 3 Similar to Part 1, except that when $\Theta_2 = \bullet$, we apply rule **A-CTXE-KUVAR****SOLVED**. □

Lemma D.25 (Parallel Extension Solution). *If $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$, and $[\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$, then $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$.*

PROOF. By induction on Θ_2 .

- Case Θ_2 is empty. Then Δ_2 must be empty. Then $\Delta_1, \widehat{\alpha} \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2$. By inversion we have $\Delta_1 \longrightarrow \Theta_1$. And $\Delta_1, \widehat{\alpha} = \kappa_1 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2$ holds by rule **A-CTXE-KUVAR****SOLVED**.
- Case $\Theta_2 = \Theta'_2, a : \kappa$. Then $\Delta_2 = \Delta'_2, a : \kappa$. By I.H., we $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta'_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2$. Then by rule **A-TCTXE-TVAR** we are done.
- Case $\Theta_2 = \Theta'_2, T : \kappa$. By I.H. and rule **A-TCTXE-TCON**.

- Case $\Theta_2 = \Theta'_2, \widehat{\alpha}_2$. Then the derivation of $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2, \widehat{\alpha}_2$ must conclude with either rule **A-CTXE-KUVAR** or rule **A-CTXE-ADD**.
 - Subcase: the derivation concludes with rule **A-CTXE-KUVAR**. It must be $\Delta_2 = \Delta'_2, \widehat{\alpha}_1$.

$$\begin{array}{l|l} \Delta_1, \widehat{\alpha}, \Delta'_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2 & \text{Given} \\ \Delta_1, \widehat{\alpha} = \kappa_1, \Delta'_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2 & \text{I.H.} \\ \Delta_1, \widehat{\alpha} = \kappa_1, \Delta'_2, \widehat{\alpha}_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2, \widehat{\alpha}_2 & \text{By rule A-CTXE-KUVAR} \end{array}$$

- Subcase: the derivation concludes with rule **A-CTXE-ADD**.

$$\begin{array}{l|l} \Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2 & \text{Given} \\ \Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2 & \text{I.H.} \\ \Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta'_2, \widehat{\alpha}_1 & \text{By rule A-CTXE-ADD} \end{array}$$

- Case $\Theta_2 = \Theta_2, \widehat{\alpha}_2 = \kappa$. This case is similar to the last one. □

Lemma D.26 (Parallel Variable Update). *If $\Delta_1, \widehat{\alpha}, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa, \Theta_2$, and $\Delta_1 \Vdash^{\text{kv}} \kappa_1$, and $\Theta_1 \Vdash^{\text{kv}} \kappa_2$, and $[\Theta_1]\kappa = [\Theta_1]\kappa_1 = [\Theta_1]\kappa_2$, then $\Delta_1, \widehat{\alpha} = \kappa_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} = \kappa_2, \Theta_2$*

PROOF. The proof is exactly the same as the one for Lemma D.25. Except for the case when Θ_2 is empty, we use rule **A-CTXE-SOLVE**. □

D.2.4 Properties of Complete Context.

Lemma D.27 (Type Constructor Preservation). *If Δ ok, and $(T : \kappa) \in \Delta$, and $\Delta \longrightarrow \Omega$, then $(T : [\Omega]\kappa) \in [\Omega]\Delta$.*

PROOF. Suppose $\Delta = \Delta_1, T : \kappa, \Delta_2$. Then by Lemma D.17 we know $\Omega = \Omega_1, T : \kappa, \Omega_2, \Delta_1 \longrightarrow \Omega_1$. So $(T : [\Omega_1]\kappa) \in [\Omega_1]\Delta$ according to the definition of context application. Because Δ ok, and $\Delta \longrightarrow \Omega$, by Lemma D.15 we have Ω ok. So by inversion we have $\Omega_1 \Vdash^{\text{kv}} \kappa$. By Lemma D.19 we have $[\Omega]\kappa = [\Omega_1]\kappa$. Therefore $(T : [\Omega]\kappa) \in [\Omega]\Delta$. □

Lemma D.28 (Type Variable Preservation). *If Δ ok, and $(a : \kappa) \in \Delta$, and $\Delta \longrightarrow \Omega$, then $(a : [\Omega]\kappa) \in [\Omega]\Delta$.*

PROOF. This lemma is similar to Lemma D.27. □

Lemma D.29 (Finishing Kinding). *If Ω ok, and $\Omega \Vdash^{\text{kv}} \kappa$, and $\Omega \longrightarrow \Omega'$, then $[\Omega]\kappa = [\Omega']\kappa$.*

PROOF. By Lemma D.18 we know $[\Omega']\kappa = [\Omega']([\Omega]\kappa)$. Because $[\Omega]\kappa$ contains no unsolved kind unification variable, we have $[\Omega']([\Omega]\kappa) = [\Omega]\kappa$. Therefore $[\Omega']\kappa = [\Omega]\kappa$. □

Lemma D.30 (Finishing Term Contexts). *If Ω ok, and $\Omega \Vdash^{\text{lectx}} \Gamma$, and $\Omega \longrightarrow \Omega'$, then $[\Omega']\Gamma = [\Omega]\Gamma$.*

PROOF. By $\Omega \Vdash^{\text{lectx}} \Gamma$, we have that any kind κ that appears in Γ has $\Omega \Vdash^{\text{kv}} \kappa$. So our goal follows directly from Lemma D.29. □

Lemma D.31 (Stability of Complete Contexts). *If $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta = [\Omega]\Omega$.*

PROOF. By induction on $\Delta \longrightarrow \Omega$.

- Case

$$\frac{\text{A-CTXE-EMPTY}}{\bullet \longrightarrow \bullet}$$

The goal follows trivially.

- Case

$$\frac{\text{A-CTXE-TVAR} \quad \Delta \longrightarrow \Theta}{\Delta, a : \kappa \longrightarrow \Theta, a : \kappa}$$

We have $\Delta = \Delta', a : \kappa$, $\Omega = \Omega', a : \kappa$ and $\Delta' \longrightarrow \Omega'$.

$$\begin{array}{l|l} [\Omega]\Delta & \\ = [\Omega', a : \kappa](\Delta', a : \kappa) & \text{By definition} \\ = [\Omega']\Delta', a : [\Omega']\kappa & \text{By definition} \\ = [\Omega']\Omega', a : [\Omega']\kappa & \text{By I.H.} \\ = \Omega', a : \kappa & \text{By definition} \end{array}$$

- Case

$$\frac{\text{A-CTXE-TCON} \quad \Delta \longrightarrow \Theta}{\Delta, T : \kappa \longrightarrow \Theta, T : \kappa}$$

This case is similar to the case for rule **A-CTXE-TVAR**.

- Case

$$\frac{\text{A-CTXE-KUVAR} \quad \Delta \longrightarrow \Theta}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha}}$$

This case is impossible as Ω is a complete context.

- Case

$$\frac{\text{A-CTXE-KUVAR}^{\text{SOLVED}} \quad \Delta \longrightarrow \Theta \quad [\Theta]\kappa_1 = [\Theta]\kappa_2}{\Delta, \widehat{\alpha} = \kappa_1 \longrightarrow \Theta, \widehat{\alpha} = \kappa_2}$$

We have $\Delta = \Delta', \widehat{\alpha} = \kappa_1$, $\Omega = \Omega', \widehat{\alpha} = \kappa_2$ and $[\Omega']\kappa_1 = [\Omega']\kappa_2$.

$$\begin{array}{l|l} [\Omega]\Delta & \\ = [\Omega', \widehat{\alpha} = \kappa_2](\Delta', \widehat{\alpha} = \kappa_1) & \text{By definition} \\ = [\Omega']\Delta' & \text{By definition} \\ = [\Omega']\Omega' & \text{By I.H.} \\ = \Omega', \widehat{\alpha} = \kappa_2 & \text{By definition} \end{array}$$

- Case

$$\frac{\text{A-CTXE-SOLVE} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta, \widehat{\alpha} \longrightarrow \Theta, \widehat{\alpha} = \kappa}$$

This case is similar to the case for rule **A-CTXE-KUVAR**^{SOLVED}.

- Case

$$\frac{\text{A-CTXE-ADD} \quad \Delta \longrightarrow \Theta}{\Delta \longrightarrow \Theta, \widehat{\alpha}}$$

This case is impossible as Ω is a complete context.

- Case

$$\frac{\text{A-CTXE-ADDSOLVED} \quad \Delta \longrightarrow \Theta \quad \Theta \Vdash^{\text{kv}} \kappa}{\Delta \longrightarrow \Theta, \widehat{\alpha} = \kappa}$$

This case is similar to the case for rule **A-CTXE-KUVAR**SOLVED.

□

Lemma D.32 (Softness Goes Away). *If $\Delta_1, \Delta_2 \longrightarrow \Omega_1, \Omega_2$ where $\Delta_1 \longrightarrow \Omega_1$, and Δ_2 soft, then $[\Omega_1, \Omega_2](\Delta_1, \Delta_2) = [\Omega_1]\Delta_1$.*

PROOF. By induction on Δ_2 and the goal follows directly from the definition of context application.

□

Lemma D.33 (Confluence of Completeness). *If $\Delta_1 \longrightarrow \Omega$, and $\Delta_2 \longrightarrow \Omega$, then $[\Omega]\Delta_1 = [\Omega]\Delta_2$.*

PROOF. By Lemma D.31 we have $[\Omega]\Delta_1 = [\Omega]\Omega$ and $[\Omega]\Delta_2 = [\Omega]\Omega$. Therefore $[\Omega]\Delta_1 = [\Omega]\Delta_2$.

□

Lemma D.34 (Finishing Completions). *If Ω ok, and $\Omega \longrightarrow \Omega'$, then $[\Omega]\Omega = [\Omega']\Omega'$.*

PROOF. By induction on $\Omega \longrightarrow \Omega'$.

- Case

$$\frac{\text{A-CTXE-EMPTY}}{\bullet \longrightarrow \bullet}$$

The goal follows trivially.

- Cases for rules **A-CTXE-KUVAR** and **A-CTXE-ADD** are impossible as Ω and Ω' are complete contexts.
- Case

$$\frac{\text{A-CTXE-TVAR} \quad \Delta \longrightarrow \Theta}{\Delta, a : \kappa \longrightarrow \Theta, a : \kappa}$$

So we have $\Omega = \Omega_1, a : \kappa$, and $\Omega' = \Omega'_1, a : \kappa$.

$[\Omega_1]\Omega_1 = [\Omega'_1]\Omega'_1$ $\Omega_1, a : \kappa = [\Omega_1]\Omega_1, a : [\Omega_1]\kappa$ $\Omega'_1, a : \kappa = [\Omega'_1]\Omega'_1, a : [\Omega'_1]\kappa$ Ω ok Ω_1 ok \wedge $\Omega_1 \Vdash^{\text{kv}} \kappa$ $[\Omega_1]\kappa = [\Omega'_1]\kappa$ $\Omega_1, a : \kappa = \Omega'_1, a : \kappa$	By I.H. By definition By definition Given By inversion By Lemma D.29 Follows from the equations
---	---

- The rest cases are similar to the above case.

□

D.2.5 Soundness of Algorithm.

Lemma D.35 (Soundness of Kind Validating). *If Ω ok, and $\Omega \Vdash^{\text{kv}} \kappa$, then $[\Omega]\kappa$ is a validate kind in the declarative system.*

PROOF. By induction on the size of $|\Omega \vdash \kappa|$. Then case analyze on κ .

- Case $\kappa = \star$. Follows trivially by $[\Omega]\star = \star$.

- Case $\kappa = \kappa_1 \rightarrow \kappa_2$. Follows directly from I.H..
- Case $\kappa = \widehat{\alpha}$. Ω must be $\Omega_1, \widehat{\alpha} = \kappa, \Omega_2$, and $[\Omega]\widehat{\alpha} = [\Omega]\kappa$. By I.H., we know $[\Omega]\kappa$ is a well-formed kind.

□

Lemma D.36 (Soundness of Well-formed Type Context). *If Δ ok, and $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta$ is a valid type context in the declarative system.*

PROOF. By induction on the well-formedness of type context.

- Case

$$\text{A-TCTX-EMPTY}$$

$$\frac{}{\bullet \text{ ok}}$$

Holds trivially.

- Case

$$\frac{\text{A-TCTX-TVAR} \quad \Delta \text{ ok} \quad \Delta \Vdash^{\text{kv}} \kappa}{\Delta, a : \kappa \text{ ok}}$$

$\Delta, a : \kappa \longrightarrow \Omega$ $\Omega = \Omega_1, a : \kappa, \Omega_2 \wedge \Omega_2 \text{ soft} \wedge \Delta \longrightarrow \Omega_1$ $[\Omega](\Delta, a : \kappa)$ $= [\Omega_1, a : \kappa, \Omega_2](\Delta, a : \kappa)$ $= [\Omega_1, a : \kappa](\Delta, a : \kappa)$ $= [\Omega_1]\Delta, a : [\Omega_1]\kappa$ $[\Omega_1]\Delta$ is a valid declarative type context $[\Omega_1]\kappa$ is a declarative validate kind $[\Omega_1]\Delta, a : [\Omega_1]\kappa$ is a valid type context	Given By Lemma D.17 By Lemma D.32 By definition I.H. By Lemma D.35
---	---

- Case

$$\frac{\text{A-TCTX-TCON} \quad \Delta \text{ ok} \quad \Delta \Vdash^{\text{kv}} \kappa}{\Delta, T : \kappa \text{ ok}}$$

This case is similar to the case rule **A-TCTX-TVAR**.

- Case

$$\frac{\text{A-TCTX-KUVAR} \quad \Delta \text{ ok}}{\Delta, \widehat{\alpha} \text{ ok}}$$

$\Delta, \widehat{\alpha} \longrightarrow \Omega$ $\Omega = \Omega_1, \widehat{\alpha} = \kappa, \Omega_2 \wedge \Omega_2 \text{ soft} \wedge \Delta \longrightarrow \Omega_1$ $[\Omega](\Delta, \widehat{\alpha})$ $= [\Omega_1, \widehat{\alpha} = \kappa, \Omega_2](\Delta, \widehat{\alpha})$ $= [\Omega_1]\Delta$ $[\Omega_1]\Delta$ is a valid declarative type context	Given By Lemma D.17 By Lemma D.32 I.H.
--	---

- Case

$$\frac{\text{A-TCTX-KUVAR SOLVED} \quad \Delta \text{ ok} \quad \Delta \Vdash^{\text{kv}} \kappa}{\Delta, \widehat{\alpha} = \kappa \text{ ok}}$$

This case is similar to the case rule **A-TCTX-KUVAR**.

□

Lemma D.37 (Soundness of Well-formed Term Context). *If Δ ok, and $\Delta \Vdash^{\text{ectx}} \Gamma$, and $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta \vdash [\Omega]\Gamma$.*

PROOF. By induction on the judgment of well-formed term context.

- Case

$$\frac{\text{A-ECTX-EMPTY}}{\Delta \Vdash^{\text{ectx}} \bullet}$$

Follows trivially by rule **ECTX-EMPTY**.

- Case

$$\frac{\text{A-ECTX-VAR} \quad \Delta \Vdash^{\text{ectx}} \Gamma \quad \Delta \Vdash^{\text{kc}} \sigma \Leftarrow \star}{\Delta \Vdash^{\text{ectx}} \Gamma, x : \sigma}$$

$[\Omega]\Delta \vdash [\Omega]\Gamma$ $[\Omega]\Delta \Vdash^{\text{k}} [\Omega]\sigma : [\Omega]\star$ $[\Omega]\Delta \Vdash^{\text{k}} [\Omega]\sigma : \star$ $[\Omega](\Gamma, x : \sigma) = [\Omega]\Gamma, x : [\Omega]\sigma$ $[\Omega]\Delta \vdash [\Omega](\Gamma, x : \sigma)$	I.H. By Lemma D.41 By definition By definition By rule ECTX-VAR
---	--

- Case

$$\frac{\text{A-ECTX-DCON} \quad \Delta \Vdash^{\text{ectx}} \Gamma \quad \Delta \Vdash^{\text{kc}} \sigma \Leftarrow \star}{\Delta \Vdash^{\text{ectx}} \Gamma, D : \sigma}$$

This case is similar to the case for rule **A-ECTX-VAR**.

□

Lemma D.38 (Soundness of Promotion). *If Δ ok, and $\Delta \Vdash_{\alpha}^{\text{pr}} \kappa_1 \rightsquigarrow \kappa_2 \dashv \Theta$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2$. Moreover, if $\Delta \Vdash^{\text{kv}} \kappa_1$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\kappa_1 = [\Omega]\kappa_2$.*

PROOF. By Lemma D.3 and Lemma D.18, if given $[\Theta]\kappa_1 = [\Theta]\kappa_2$, we can prove $[\Omega]\kappa_1 = [\Omega]\kappa_2$. Thus we only need to prove that $[\Theta]\kappa_1 = [\Theta]\kappa_2$.

By a straightforward induction on the promotion judgment. All cases follow trivially.

□

Lemma D.39 (Soundness of Unification). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa_1$, and $\Delta \Vdash^{\text{kv}} \kappa_2$, and $\Delta \Vdash^{\mu} \kappa_1 \approx \kappa_2 \dashv \Theta$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\kappa_1 = [\Omega]\kappa_2$.*

PROOF. By Lemma D.4, Lemma D.10 and Lemma D.18, if given $[\Theta]\kappa_1 = [\Theta]\kappa_2$. we can prove $[\Omega]\kappa_1 = [\Omega]\kappa_2$. Thus we only need to prove that $[\Theta]\kappa_1 = [\Theta]\kappa_2$.

By induction on the unification judgment.

- Case

$$\frac{\text{A-U-REFL}}{\Delta \Vdash^{\mu} \kappa \approx \kappa \dashv \Delta}$$

$$[\Delta]\kappa = [\Delta]\kappa.$$

- Case

$$\frac{\text{A-U-ARROW} \quad \Delta \Vdash^{\mu} \kappa_1 \approx \kappa_3 \dashv \Theta_1 \quad \Theta_1 \Vdash^{\mu} [\Theta_1]\kappa_2 \approx [\Theta_1]\kappa_4 \dashv \Theta}{\Delta \Vdash^{\mu} \kappa_1 \rightarrow \kappa_2 \approx \kappa_3 \rightarrow \kappa_4 \dashv \Theta}$$

$\Delta \rightarrow \Theta_1$	By Lemma D.4
$\Theta_1 \rightarrow \Theta$	By Lemma D.4
$\Delta \rightarrow \Theta$	By Lemma D.20
$[\Theta_1]\kappa_1 = [\Theta_1]\kappa_3$	By I.H.
$[\Theta]\kappa_1 = [\Theta]\kappa_3$	By Lemma D.18
$\Delta \Vdash^{\text{kv}} \kappa_2$	By inversion
$\Theta_1 \Vdash^{\text{kv}} [\Theta_1]\kappa_2$	By Lemma D.10 and Lemma D.12
$\Theta_1 \Vdash^{\text{kv}} [\Theta_1]\kappa_4$	As above
$[\Theta]\kappa_2 = [\Theta]\kappa_4$	By I.H. and Lemma D.18
$[\Theta](\kappa_1 \rightarrow \kappa_2) = [\Theta](\kappa_3 \rightarrow \kappa_4)$	Follows directly

- Case

$$\frac{\text{A-U-KVARL} \quad \Delta \Vdash_{\hat{\alpha}}^{\text{pr}} \kappa \rightsquigarrow \kappa_2 \dashv \Theta[\hat{\alpha}]}{\Delta[\hat{\alpha}] \Vdash^{\mu} \hat{\alpha} \approx \kappa \dashv \Theta[\hat{\alpha} = \kappa_2]}$$

$[\Theta[\hat{\alpha}]]\kappa = [\Theta[\hat{\alpha}]]\kappa_2$	By Lemma D.38
$\Delta \rightarrow \Theta[\hat{\alpha}] \wedge \Theta[\hat{\alpha}] \Vdash^{\text{kv}} \kappa_2$	By Lemma D.3
$\Theta[\hat{\alpha}] \rightarrow \Theta[\hat{\alpha} = \kappa_2]$	By Lemma D.3 and Lemma D.21
$[\Theta[\hat{\alpha} = \kappa_2]]\kappa = [\Theta[\hat{\alpha} = \kappa_2]]\kappa_2$	By Lemma D.18

- Case

$$\frac{\text{A-U-KVARR} \quad \Delta \Vdash_{\hat{\alpha}}^{\text{pr}} \kappa \rightsquigarrow \kappa_2 \dashv \Theta[\hat{\alpha}]}{\Delta[\hat{\alpha}] \Vdash^{\mu} \kappa \approx \hat{\alpha} \dashv \Theta[\hat{\alpha} = \kappa_2]}$$

This case is similar to the case for rule **A-U-KVARL**.

□

Lemma D.40 (Soundness of Application Kinding). *If Δ ok, and $\Delta \Vdash^{\text{kv}} \kappa_1$, and $\Delta \Vdash^{\text{kv}} \kappa_2$, and $\Delta \Vdash^{\text{kapp}} \kappa_1 \bullet \kappa_2 : \kappa_3 \dashv \Theta$, then $[\Theta]\kappa_1 = [\Theta]\kappa_2 \rightarrow [\Theta]\kappa_3$. If $\Theta \rightarrow \Omega$, then $[\Omega]\kappa_1 = [\Omega]\kappa_2 \rightarrow [\Omega]\kappa_3$.*

PROOF. By Lemma D.5, Lemma D.10 and Lemma D.18, we know $[\Omega]\kappa_1 = [\Omega]([\Theta]\kappa_1)$ and $[\Omega]\kappa_2 = [\Omega]([\Theta]\kappa_2)$ and $[\Omega]\kappa_3 = [\Omega]([\Theta]\kappa_3)$. Thus we only need to prove that $[\Theta]\kappa_1 = [\Theta]\kappa_2 \rightarrow [\Theta]\kappa_3$.

By induction on the application kinding judgment.

- Case

$$\frac{\text{A-KAPP-KUVAR} \quad \Delta[\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha} = \hat{\alpha}_1 \rightarrow \hat{\alpha}_2] \Vdash^{\mu} \hat{\alpha}_1 \approx \kappa \dashv \Theta}{\Delta[\hat{\alpha}] \Vdash^{\text{kapp}} \hat{\alpha} \bullet \kappa : \hat{\alpha}_2 \dashv \Theta}$$

$[\Theta]\hat{\alpha}_1 = [\Theta]\kappa$	By Lemma D.39
$\Delta[\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha} = \hat{\alpha}_1 \rightarrow \hat{\alpha}_2] \rightarrow \Theta$	By Lemma D.4
$[\Theta]\hat{\alpha}$	
$= [\Theta]([\Delta[\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha} = \hat{\alpha}_1 \rightarrow \hat{\alpha}_2]]\hat{\alpha})$	By Lemma D.18
$= [\Theta]([\Delta[\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha} = \hat{\alpha}_1 \rightarrow \hat{\alpha}_2]]\hat{\alpha}_1 \rightarrow \hat{\alpha}_2)$	By definition

$$\begin{array}{l}
= [\Theta](\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2) \\
= [\Theta]\widehat{\alpha}_1 \rightarrow [\Theta]\widehat{\alpha}_2 \\
= [\Theta]\kappa \rightarrow [\Theta]\widehat{\alpha}_2
\end{array}
\left| \begin{array}{l}
\text{By Lemma D.18} \\
\text{By definition} \\
\text{By substituting the equation}
\end{array} \right.$$

- Case

$$\frac{\text{A-KAPP-ARROW} \quad \Delta \Vdash^{\text{M}} \kappa_1 \approx \kappa \dashv \Theta}{\Delta \Vdash^{\text{kapp}} \kappa_1 \rightarrow \kappa_2 \bullet \kappa : \kappa_2 \dashv \Theta}$$

$$\begin{array}{l}
[\Theta]\kappa_1 = [\Theta]\kappa \\
[\Theta](\kappa_1 \rightarrow \kappa_2) \\
= [\Theta]\kappa_1 \rightarrow [\Theta]\kappa_2 \\
= [\Theta]\kappa \rightarrow [\Theta]\kappa_2
\end{array}
\left| \begin{array}{l}
\text{By Lemma D.39} \\
\text{By definition} \\
\text{By substituting the equation}
\end{array} \right.$$

□

Lemma D.41 (Soundness of Kinding). *If Δ ok, and $\Delta \Vdash^{\text{k}} \sigma : \kappa \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{k}} [\Omega]\sigma : [\Omega]\kappa$.*

PROOF. By induction on the kinding judgment.

- Case

$$\frac{\text{A-K-NAT}}{\Delta \Vdash^{\text{k}} \text{Int} : \star \dashv \Delta}$$

$$[\Omega]\Delta \Vdash^{\text{k}} [\Omega]\text{Int} : [\Omega]\star \text{ By rule K-NAT.}$$

- Case

$$\frac{\text{A-K-VAR} \quad (a : \kappa) \in \Delta}{\Delta \Vdash^{\text{k}} a : \kappa \dashv \Delta}$$

$$\begin{array}{l}
(a : \kappa) \in \Delta \\
(a : [\Omega]\kappa) \in [\Omega]\Delta \\
[\Omega]\Delta \vdash [\Omega]a : [\Omega]\kappa
\end{array}
\left| \begin{array}{l}
\text{Given} \\
\text{By Lemma D.28} \\
\text{By rule K-VAR}
\end{array} \right.$$

- Case

$$\frac{\text{A-K-TCON} \quad (T : \kappa) \in \Delta}{\Delta \Vdash^{\text{k}} T : \kappa \dashv \Delta}$$

Similar to the rule **A-K-VAR** case with Lemma D.27.

- Case

$$\frac{\text{A-K-ARROW}}{\Delta \Vdash^{\text{k}} \rightarrow : \star \rightarrow \star \rightarrow \star \dashv \Delta}$$

$$[\Omega]\Delta \Vdash^{\text{k}} [\Omega] \rightarrow : [\Omega](\star \rightarrow \star \rightarrow \star) \text{ by rule K-ARROW.}$$

- Case

$$\frac{\text{A-K-FORALL} \quad \Delta \Vdash^{\text{kv}} \kappa \quad \Delta, a : \kappa \Vdash^{\text{k}} \sigma : \kappa_2 \dashv \Theta, a : \kappa \quad [\Theta]\kappa_2 = \star}{\Delta \Vdash^{\text{k}} \forall a : \kappa. \sigma : \star \dashv \Theta}$$

$$\begin{array}{l}
\Theta \longrightarrow \Omega \\
\Theta, a : \kappa \longrightarrow \Omega, a : \kappa
\end{array}
\left| \begin{array}{l}
\text{Given} \\
\text{By rule A-CTXE-TVAR}
\end{array} \right.$$

$ \begin{array}{l} [\Omega, a : \kappa](\Delta, a : \kappa) \Vdash^k [\Omega, a : \kappa]\sigma : [\Omega, a : \kappa]\kappa_2 \\ [\Omega]\Delta, a : [\Omega]\kappa \Vdash^k [\Omega, a : \kappa]\sigma : [\Omega, a : \kappa](\Theta, a : \kappa)\kappa_2 \\ [\Omega]\Delta, a : [\Omega]\kappa \Vdash^k [\Omega, a : \kappa]\sigma : [\Omega, a : \kappa]\star \\ [\Omega]\Delta, a : [\Omega]\kappa \Vdash^k [\Omega]\sigma : [\Omega]\star \\ [\Omega]\Delta \Vdash^k \forall a : [\Omega]\kappa. [\Omega]\sigma : [\Omega]\star \end{array} $	<table style="border-collapse: collapse; width: 100%;"> <tr><td style="border-right: 1px solid black; padding-right: 5px;">I.H.</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.18</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By definition</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By property of context application</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By rule K-FORALL</td></tr> </table>	I.H.	By Lemma D.18	By definition	By property of context application	By rule K-FORALL
I.H.						
By Lemma D.18						
By definition						
By property of context application						
By rule K-FORALL						

• Case

$$\frac{\text{A-K-APP} \quad \Delta \Vdash^k \tau_1 : \kappa_1 \dashv \Theta_1 \quad \Theta_1 \Vdash^k \tau_2 : \kappa_2 \dashv \Theta_2 \quad \Theta_2 \Vdash^{\text{kapp}} [\Theta_2]\kappa_1 \bullet [\Theta_2]\kappa_2 : \kappa_3 \dashv \Theta}{\Delta \Vdash^k \tau_1 \tau_2 : \kappa_3 \dashv \Theta}$$

$ \begin{array}{l} \Theta_1 \longrightarrow \Theta_2 \\ \Theta_2 \longrightarrow \Theta \\ \Theta_1 \longrightarrow \Omega \\ [\Omega]\Delta \Vdash^k [\Omega]\tau_1 : [\Omega]\kappa_1 \\ [\Omega]\Delta \Vdash^k [\Omega]\tau_2 : [\Omega]\kappa_2 \\ [\Omega]\kappa_1 = [\Omega](\kappa_2 \rightarrow \kappa_3) = [\Omega]\kappa_2 \rightarrow [\Omega]\kappa_3 \\ [\Omega]\Delta \Vdash^k [\Omega](\tau_1 \tau_2) : [\Omega]\kappa_3 \end{array} $	<table style="border-collapse: collapse; width: 100%;"> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.6</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.4</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.20</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">I.H.</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">Similarly</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.40</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By rule K-APP-P</td></tr> </table>	By Lemma D.6	By Lemma D.4	By Lemma D.20	I.H.	Similarly	By Lemma D.40	By rule K-APP-P
By Lemma D.6								
By Lemma D.4								
By Lemma D.20								
I.H.								
Similarly								
By Lemma D.40								
By rule K-APP-P								

□

Lemma D.42 (Soundness of Typing Data Constructor Declaration). *If Δ ok, and $\Delta \Vdash_{\tau}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash_{\tau}^{\text{dc}} \mathcal{D} \rightsquigarrow \tau$.*

PROOF. We have

$$\frac{\text{A-DC-DECL} \quad \Delta \Vdash^k \overline{\tau}_i^i \rightarrow \tau : \star \dashv \Theta}{\Delta \Vdash_{\tau}^{\text{dc}} D \overline{\tau}_i^i \rightsquigarrow \overline{\tau}_i^i \rightarrow \tau \dashv \Theta}$$

Follows directly from Lemma D.41 and rule **DC-DECL**.

□

Lemma D.43 (Soundness of Typing Datatype Declaration). *If Δ ok, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow [\Omega]\Gamma$.*

PROOF. We have

A-DT-DECL

$$\frac{(T : \kappa) \in \Delta \quad \Delta, \overline{\alpha}_i^i \Vdash^{\mu} [\Delta]\kappa \approx (\overline{\alpha}_i^i \rightarrow \star) \dashv \Theta_1, \overline{\alpha}_i = \kappa_i \quad \overline{\Theta}_j, \overline{a}_i : \kappa_i^i \Vdash_{T \overline{a}_i}^{\text{dc}} \mathcal{D}_j \rightsquigarrow \tau_j \dashv \Theta_{j+1}, \overline{a}_i : \kappa_i^i}{\Delta \Vdash^{\text{dt}} \text{data } T \overline{a}_i^i = \overline{\mathcal{D}}_j^{j \in 1..n} \rightsquigarrow \overline{D}_j : \forall \overline{a}_i : \kappa_i^i. \tau_j \dashv \Theta_{n+1}}$$

$ \begin{array}{l} \Delta, \overline{\alpha}_i^i \longrightarrow \Theta_1, \overline{\alpha}_i = \kappa_i \\ \Delta \longrightarrow \Theta_1 \\ \overline{\Theta}_j, \overline{a}_i : \kappa_i^i \longrightarrow \overline{\Theta}_{j+1}, \overline{a}_i : \kappa_i^i \\ \overline{\Theta}_j \longrightarrow \overline{\Theta}_{j+1}^j \\ \Theta_{n+1} \longrightarrow \Omega \\ \overline{\Theta}_j \longrightarrow \Omega^j \\ \Delta \longrightarrow \Omega \\ \Theta_{n+1}, \overline{a}_i : \kappa_i^i \longrightarrow \Omega, \overline{a}_i : \kappa_i^i \end{array} $	<table style="border-collapse: collapse; width: 100%;"> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.4</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.17</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.7</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.17</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">Given</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.20</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By Lemma D.20</td></tr> <tr><td style="border-right: 1px solid black; padding-right: 5px;">By rule A-CTXE-TVAR</td></tr> </table>	By Lemma D.4	By Lemma D.17	By Lemma D.7	By Lemma D.17	Given	By Lemma D.20	By Lemma D.20	By rule A-CTXE-TVAR
By Lemma D.4									
By Lemma D.17									
By Lemma D.7									
By Lemma D.17									
Given									
By Lemma D.20									
By Lemma D.20									
By rule A-CTXE-TVAR									

$\frac{\overline{\Theta_{j+1}, \overline{a_i : \kappa_i^i} \longrightarrow \Omega, \overline{a_i : \kappa_i^i}^j}}{\overline{[\Omega, \overline{a_i : \kappa_i^i}](\Theta_{j+1}, \overline{a_i : \kappa_i^i}) \stackrel{\text{dc}}{T \overline{a_i}} \mathcal{D}_j \rightsquigarrow [\Omega] \tau_j^j}}$	By Lemma D.20
$\overline{[\Omega] \Theta_{j+1}, \overline{a_i : [\Omega] \kappa_i^i} \stackrel{\text{dc}}{T \overline{a_i}} \mathcal{D}_j \rightsquigarrow [\Omega] \tau_j^j}$	By Lemma D.42
$[\Omega] \Delta, \overline{a_i : [\Omega] \kappa_i^i} \stackrel{\text{dc}}{T \overline{a_i}} \mathcal{D}_j \rightsquigarrow [\Omega] \tau_j^j$	By definition
$(T : \kappa) \in \Delta$	(1) By Lemma D.33
$(T : [\Omega] \kappa) \in [\Omega] \Delta$	Given
$\Theta_1, \overline{\widehat{\alpha}_i = \kappa_i^i} \longrightarrow \Omega, \overline{\widehat{\alpha}_i = \kappa_i^i}$	(2) By Lemma D.27
$[\Omega, \overline{\widehat{\alpha}_i = \kappa_i^i}](\Delta \kappa) = [\Omega, \overline{\widehat{\alpha}_i = \kappa_i^i}](\widehat{\alpha}_i^i \rightarrow \star)$	By rule A-CTXE-KUVAR SOLVED
$[\Omega](\Delta \kappa) = \overline{[\Omega] \kappa_i^i} \rightarrow \star$	By Lemma D.39
$[\Omega] \kappa = \overline{[\Omega] \kappa_i^i} \rightarrow \star$	By definition
$[\Omega] \Delta \stackrel{\text{dt}}{\rightsquigarrow} [\Omega] \Gamma$	(3) By Lemma D.18
	By rule DT-DECL and (1), (2), (3)

□

Lemma D.44 (Soundness of Typing Program). *If Ω ok, and $\Omega \Vdash^{\text{ectx}} \Gamma$, and $\Omega; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$, then $[\Omega] \Omega; [\Omega] \Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$.*

PROOF. By induction on the typing program judgment.

- Case

$$\frac{\text{A-PGM-EXPR} \quad [\Omega] \Omega; [\Omega] \Gamma \vdash e : \sigma}{\Omega; \Gamma \Vdash^{\text{pgm}} e : \sigma}$$

The conclusion holds directly from the hypothesis and rule PGM-EXPR.

- Case

$$\frac{\text{A-PGM-DT} \quad \Theta_1 = \Omega, \overline{\widehat{\alpha}_i^i}, \overline{T_i : \widehat{\alpha}_i^i} \quad \Theta_{n+1} \longrightarrow \Omega' \quad \Omega'; \Gamma, \overline{\widehat{\alpha}_i^i} \Vdash^{\text{pgm}} \text{pgm} : \sigma}{\Omega; \Gamma \Vdash^{\text{pgm}} \text{rec } \overline{\mathcal{T}_i^{i \in 1..n}}; \text{pgm} : \sigma}$$

$\overline{\Theta_i \longrightarrow \Theta_{i+1}}$	By Lemma D.8
$\overline{\Theta_{n+1} \longrightarrow \Omega'}$	By Lemma D.13
$\overline{\Theta_i \longrightarrow \Omega'^i}$	By Lemma D.20
$\Omega, \overline{\widehat{\alpha}_i^{i \in 1..n}}, \overline{T_i : \widehat{\alpha}_i^{i \in 1..n}} \longrightarrow \Omega'$	$\Theta_1 = \Omega, \overline{\widehat{\alpha}_i^{i \in 1..n}}, \overline{T_i : \widehat{\alpha}_i^{i \in 1..n}}$
$\Omega' = \Omega'', \overline{\widehat{\alpha}_i = \kappa_i, \Omega_i^{i \in 1..n}}, \overline{T_i : \widehat{\alpha}_i, \Omega_i'^{i \in 1..n}}$	By Lemma D.17
$\overline{\Omega_i \text{ soft}^i \wedge \Omega_i' \text{ soft}^i}$	Above
$\Omega \longrightarrow \Omega''$	Above
$[\Omega'] \Omega'$	
$= [\Omega'', \overline{\widehat{\alpha}_i = \kappa_i, \Omega_i^{i \in 1..n}}, \overline{T_i : \widehat{\alpha}_i, \Omega_i'^{i \in 1..n}}]$	
$(\Omega'', \overline{\widehat{\alpha}_i = \kappa_i, \Omega_i^{i \in 1..n}}, \overline{T_i : \widehat{\alpha}_i, \Omega_i'^{i \in 1..n}})$	
$= [\Omega''] \Omega'', \overline{T_i : \kappa_i^{i \in 1..n}}$	for some κ_i , by Lemma D.32
$= [\Omega] \Omega, \overline{T_i : \kappa_i^{i \in 1..n}}$	By Lemma D.34
$\overline{[\Omega'] \Theta_i \stackrel{\text{dt}}{\rightsquigarrow} [\Omega'] \Gamma_i^{i \in 1..n}}$	By Lemma D.43

$\frac{\overline{[\Omega']\Theta_i = [\Omega']\Omega'}^{i \in 1..n}}{[\Omega']\Omega' \stackrel{\mu^{\text{dt}}}{\sim} \mathcal{T}_i \sim [\Omega']\Gamma_i}^{i \in 1..n}$	By Lemma D.31
$\frac{[\Omega]\Omega, \overline{T_i : \kappa_i}^{i \in 1..n} \stackrel{\mu^{\text{dt}}}{\sim} \mathcal{T}_i \sim [\Omega']\Gamma_i}{[\Omega'](\Gamma, \overline{\Gamma_i}^{i \in 1..n}) = [\Omega']\Gamma, \overline{[\Omega']\Gamma_i}^{i \in 1..n}}$	By substituting the equation
$[\Omega']\Gamma$	(1) By substituting the equation
$= [\Omega'', \overline{\widehat{\alpha}_i = \kappa_i, \Omega_i}^{i \in 1..n}, \overline{T_i : \widehat{\alpha}_i, \Omega'_i}^{i \in 1..n}] \Gamma$	By definition
$= [\Omega''] \Gamma$	By definition and freshness
$= [\Omega] \Gamma$	By Lemma D.30
$[\Omega']\Omega'; [\Omega'](\Gamma, \overline{\Gamma_i}^{i \in 1..n}) \stackrel{\text{pgm}}{\sim} \text{pgm} : \sigma$	I.H.
$[\Omega]\Omega, \overline{T_i : \kappa_i}^{i \in 1..n}; [\Omega]\Gamma, \overline{[\Omega']\Gamma_i}^{i \in 1..n} \stackrel{\text{pgm}}{\sim} \text{pgm} : \sigma$	(2) By substituting equations
$[\Omega]\Omega; [\Omega]\Gamma \stackrel{\text{pgm}}{\sim} \text{pgm} : \sigma$	By rule PGM-DT and (1), (2)

□

D.2.6 Completeness of Algorithm.

Lemma D.45 (Completeness of Promotion). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{kv}} \widehat{\alpha}$, and $\Delta \Vdash^{\text{kv}} \kappa$, and $[\Delta]\widehat{\alpha} = \widehat{\alpha}$, and $[\Delta]\kappa = \kappa$, if κ is free of $\widehat{\alpha}$, then there exists κ_2, Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$, and $\Delta \Vdash^{\text{pr}}_{\widehat{\alpha}} \kappa \rightsquigarrow \kappa_2 \dashv \Theta$.*

PROOF. By induction on κ .

- $\kappa = \star$. Then by rule [A-PR-STAR](#), we have $\Theta = \Delta$, and $\Omega' = \Omega$.
- $\kappa = \kappa_1 \rightarrow \kappa_2$.

$\Delta \Vdash^{\text{pr}}_{\widehat{\alpha}} \kappa_1 \rightsquigarrow \kappa_3 \dashv \Delta_1 \wedge \Delta_1 \longrightarrow \Omega_1 \wedge \Omega \longrightarrow \Omega_1$	I.H.
$\Delta_1 \Vdash^{\text{kv}} \widehat{\alpha}$	By Lemma D.10
$\Delta_1 \Vdash^{\text{kv}} [\Delta_1]\kappa_2$	By Lemma D.10 and Lemma D.12
$\Delta_1 \Vdash^{\text{pr}} [\Delta_1]\kappa_2 \rightsquigarrow \kappa_4 \dashv \Theta \wedge \Theta \longrightarrow \Omega' \wedge \Omega_1 \longrightarrow \Omega'$	I.H.
$\Delta \Vdash^{\text{pr}}_{\widehat{\alpha}} \kappa_1 \rightarrow \kappa_2 \rightsquigarrow \kappa_3 \rightarrow \kappa_4 \dashv \Theta$	By rule A-PR-ARROW
$\Omega \longrightarrow \Omega'$	By Lemma D.20

- $\kappa = \widehat{\beta}$.
 - $\widehat{\beta}$ is to the left of $\widehat{\alpha}$. Then by rule [A-PR-KUVARL](#), we have $\Theta = \Delta$, and $\Omega' = \Omega$.
 - $\widehat{\beta}$ is to the right of $\widehat{\alpha}$. Then by rule [A-PR-KUVARR](#), we have $\Theta = \Delta[\widehat{\beta}_2, \widehat{\alpha}][\widehat{\beta} = \widehat{\beta}_2]$.

$\Delta[\widehat{\alpha}][\widehat{\beta}] \longrightarrow \Omega$	Given
$\Omega = \Omega[\widehat{\alpha} = \kappa_3][\widehat{\beta} = \kappa_4]$	By Lemma D.17
$\Delta[\widehat{\beta}_2, \widehat{\alpha}][\widehat{\beta}] \longrightarrow \Omega[\widehat{\beta}_2 = [\Omega]\kappa_4, \widehat{\alpha} = \kappa_3][\widehat{\beta} = \kappa_4]$	By Lemma D.24
$\Delta[\widehat{\beta}_2, \widehat{\alpha}][\widehat{\beta} = \widehat{\beta}_2] \longrightarrow \Omega[\widehat{\beta}_2 = [\Omega]\kappa_4, \widehat{\alpha} = \kappa_3][\widehat{\beta} = \kappa_4]$	By Lemma D.25
$\Omega_1 = \Omega[\widehat{\beta}_2 = [\Omega]\kappa_4, \widehat{\alpha} = \kappa_3][\widehat{\beta} = \kappa_4]$	Let

□

Lemma D.46 (Completeness of Unification). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{kv}} \kappa_1$ and $\Delta \Vdash^{\text{kv}} \kappa_2$, and $[\Delta]\kappa_1 = \kappa_1$ and $[\Delta]\kappa_2 = \kappa_2$, if $[\Omega]\kappa_1 = [\Omega]\kappa_2$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\mu} \kappa_1 \approx \kappa_2 \dashv \Theta$.*

PROOF. By case analysis on κ_1 on κ_2 .

- $\kappa_1 = \star$ and $\kappa_2 = \star$. Then by rule **A-U-REFL**, we have $\Theta = \Delta$, and $\Omega' = \Omega$.
- $\kappa_1 = \kappa_{11} \rightarrow \kappa_{12}$ and $\kappa_2 = \kappa_{21} \rightarrow \kappa_{22}$.

$ \begin{aligned} & [\Omega](\kappa_{11} \rightarrow \kappa_{12}) \\ &= [\Omega]\kappa_{11} \rightarrow [\Omega]\kappa_{12} \\ &= [\Omega](\kappa_{21} \rightarrow \kappa_{22}) \\ &= [\Omega]\kappa_{21} \rightarrow [\Omega]\kappa_{22} \\ &[\Omega]\kappa_{11} = [\Omega]\kappa_{21} \wedge [\Omega]\kappa_{12} = [\Omega]\kappa_{22} \\ &\Delta \Vdash^{\mu} \kappa_{11} \approx \kappa_{12} \dashv \Theta_1 \wedge \Theta_1 \longrightarrow \Omega_1 \wedge \Omega \longrightarrow \Omega_1 \\ &[\Omega_1](\Theta_1 \kappa_{12}) = [\Omega_1]\kappa_{12} = [\Omega_1](\kappa_{12}) \\ &= [\Omega_1](\kappa_{22}) \\ &= [\Omega_1]\kappa_{22} = [\Omega_1](\Theta_1 \kappa_{22}) \\ &\Theta_1 \Vdash^{\mu} [\Omega_1]\kappa_{21} \approx [\Omega_1]\kappa_{22} \dashv \Theta \wedge \Theta \longrightarrow \Omega' \wedge \Omega_1 \longrightarrow \Omega' \\ &\Delta \Vdash^{\mu} \kappa_{11} \rightarrow \kappa_{12} \approx \kappa_{21} \rightarrow \kappa_{22} \dashv \Theta \\ &\Omega \longrightarrow \Omega' \end{aligned} $	By definition Given By definition Follows directly (1) I.H. By Lemma D.18 Known By Lemma D.18 (2) I.H. By rule A-U-ARROW and (1) (2) By Lemma D.20
---	--

- $\kappa_2 = \widehat{\alpha}$. Then we have $[\Omega]\widehat{\alpha} = [\Omega]\kappa_1$.
 - $\kappa_1 = \widehat{\alpha}$. Then by rule **A-U-REFL**, we have $\Theta = \Delta$, and $\Omega' = \Omega$.
 - Otherwise κ_1 must be free of $\widehat{\alpha}$.

$ \begin{aligned} & \Delta \Vdash^{\text{Pr}}_{\widehat{\alpha}} \kappa_1 \rightsquigarrow \kappa_2 \dashv \Theta_1 \wedge \Theta_1 \longrightarrow \Omega' \wedge \Omega \longrightarrow \Omega' \\ & \Delta \Vdash^{\text{kv}} \widehat{\alpha} \\ & \Theta_1 = \Theta_{11}, \widehat{\alpha}, \Theta_{12} \\ & \Theta = \Theta_{11}, \widehat{\alpha} = \kappa_2, \Theta_{12} \\ & \Delta \Vdash^{\mu} \kappa_1 \approx \widehat{\alpha} \dashv \Theta \\ & [\Omega']\widehat{\alpha} \\ &= [\Omega']\kappa_1 \\ &= [\Omega']\kappa_2 \\ &\Theta \longrightarrow \Omega' \end{aligned} $	By Lemma D.45 Given By Lemma D.3 Let By rule A-U-KVARR By Lemma D.18 By Lemma D.38 By Lemma D.25
---	---

- The case when $\kappa_1 = \widehat{\alpha}$ is the same. □

Lemma D.47 (Completeness of Application Kinding). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{kv}} \kappa$ and $\Delta \Vdash^{\text{kv}} \kappa'$, and $[\Delta]\kappa = \kappa$ and $[\Delta]\kappa' = \kappa'$, if $[\Omega]\kappa = [\Omega]\kappa' \rightarrow \kappa_1$, then there exists κ_2 , Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\text{kapp}} \kappa \bullet \kappa' : \kappa_2 \dashv \Theta$, and $[\Omega']\kappa_2 = \kappa_1$.*

PROOF. By induction on κ .

- $\kappa = \widehat{\alpha}$ for some $\widehat{\alpha}$ and $[\Omega]\widehat{\alpha} = [\Omega]\kappa' \rightarrow \kappa_1$.

$ \begin{aligned} & \Delta = \Delta_1, \widehat{\alpha}, \Delta_2 \\ & \Delta_3 = \Delta_1, \widehat{\alpha}_1, \widehat{\alpha}_2, \widehat{\alpha} = \widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2, \Delta_2 \\ & \Delta \longrightarrow \Delta_3 \\ & \Omega = \Omega_1, \widehat{\alpha} = \kappa_3, \Omega_2 \\ & \Omega_3 = \Omega_1, \widehat{\alpha}_1 = [\Omega]\kappa', \widehat{\alpha}_2 = \kappa_1, \widehat{\alpha} = \kappa_3, \Omega_2 \\ & \Omega \longrightarrow \Omega_3 \\ & \Delta \longrightarrow \Omega \\ & \Delta_3 \longrightarrow \Omega_3 \\ & \Delta_3 \Vdash^{\mu} \widehat{\alpha}_1 \approx \kappa' \dashv \Theta \wedge \Theta \longrightarrow \Omega' \wedge \Omega_3 \longrightarrow \Omega' \\ & \Delta \Vdash^{\text{kapp}} \widehat{\alpha} \bullet \kappa' : \widehat{\alpha}_2 \dashv \Theta \end{aligned} $	Assume Let By Lemma D.23 , Lemma D.21 , and Lemma D.20 Assume Let By Lemma D.22 , and Lemma D.20 Given By Lemma D.24 and Lemma D.25 By Lemma D.46 By rule A-KAPP-KUVAR
--	--

$$\begin{array}{l}
\Omega \longrightarrow \Omega' \\
[\Omega']\widehat{\alpha}_2 \\
= [\Omega_3]\widehat{\alpha}_2 \\
= \kappa_1
\end{array}
\left|
\begin{array}{l}
\text{By Lemma D.20} \\
\text{By Lemma D.29}
\end{array}
\right.$$

- Case $\kappa = \kappa_{21} \rightarrow \kappa_{22}$.

$$\begin{array}{l}
[\Omega]\kappa_{21} = [\Omega]\kappa' \wedge [\Omega]\kappa_{22} = \kappa_1 \\
\Delta \Vdash^{\mu} \kappa_{21} \approx \kappa' \dashv \Theta \wedge \Theta \longrightarrow \Omega' \wedge \Omega \longrightarrow \Omega' \\
\Delta \Vdash^{\text{kapp}} \kappa \bullet \kappa' : \kappa_{22} \dashv \Theta \\
[\Omega']\kappa_{22} = [\Omega]\kappa_{22} = \kappa_1
\end{array}
\left|
\begin{array}{l}
\text{Follows directly} \\
\text{By Lemma D.46} \\
\text{By rule A-KAPP-ARROW} \\
\text{By Lemma D.29}
\end{array}
\right.$$

□

Lemma D.48 (Completeness of Kinding). *Given Δ ok and $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash^k [\Omega]\sigma : \kappa$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^k \sigma : \kappa' \dashv \Theta$, and $[\Omega']\kappa' = \kappa$.*

PROOF. By induction on the kinding judgment.

- Case

$$\frac{\text{K-NAT}}{\Sigma \Vdash^k \text{Int} : \star}$$

$$\begin{array}{l}
\Delta \Vdash^k \text{Int} : \star \dashv \Delta \\
\Theta = \Delta \\
\Omega' = \Omega
\end{array}
\left|
\begin{array}{l}
\text{By rule A-K-NAT} \\
\text{Let} \\
\text{Let}
\end{array}
\right.$$

- Case

$$\frac{\text{K-VAR} \quad (a : \kappa) \in \Sigma}{\Sigma \Vdash^k a : \kappa}$$

$$\begin{array}{l}
(a : \kappa) \in [\Omega]\Delta \\
(a : \kappa_2) \in \Delta \wedge [\Omega]\kappa_2 = \kappa \\
\Delta \Vdash^k a : \kappa_2 \dashv \Delta \\
\Theta = \Delta \\
\Omega' = \Omega
\end{array}
\left|
\begin{array}{l}
\text{Given} \\
\text{By inversion} \\
\text{By rule A-K-VAR} \\
\text{Let} \\
\text{Let}
\end{array}
\right.$$

- Case

$$\frac{\text{K-TCON} \quad (T : \kappa) \in \Sigma}{\Sigma \Vdash^k T : \kappa}$$

Similar as the case for rule **K-VAR**.

- Case

$$\frac{\text{K-ARROW}}{\Sigma \Vdash^k \rightarrow : \star \rightarrow \star \rightarrow \star}$$

Similar as the case for rule **K-NAT**.

- Case

$$\frac{\text{K-FORALL} \quad \Sigma, a : \kappa \Vdash^k \sigma : \star}{\Sigma \Vdash^k \forall a : \kappa. \sigma : \star}$$

$$\begin{aligned} & \Delta, a : \kappa \longrightarrow \Omega, a : \kappa \\ & [\Omega, a : \kappa](\Delta, a : \kappa) \Vdash^k [\Omega]\sigma : \star \\ & \Delta, a : \kappa \Vdash^k \sigma : \kappa_1 \dashv \Theta_1 \wedge \Theta_1 \longrightarrow \Omega_1 \wedge \Omega, a : \kappa \longrightarrow \Omega_1 \wedge [\Omega_1]\kappa_1 = \star \\ & \Theta_1 = \Theta, a : \kappa \\ & \Delta \Vdash^k \forall a : \kappa. \sigma : \star \dashv \Theta \\ & \Omega, a : \kappa \longrightarrow \Omega_1 \\ & \Omega_1 = \Omega_{11}, a : \kappa, \Omega_{12} \wedge \Omega \longrightarrow \Omega_{11} \\ & \Omega' = \Omega_{11} \end{aligned}$$

By rule [A-CTXE-TVAR](#)
Given
I.H.
By inversion
By rule [A-K-FORALL](#)
Known
By Lemma [D.17](#)
Let

- Case

$$\frac{\text{K-APP} \quad \Sigma \Vdash^k \tau_1 : \kappa_1 \longrightarrow \kappa_2 \quad \Sigma \Vdash^k \tau_2 : \kappa_1}{\Sigma \Vdash^k \tau_1 \tau_2 : \kappa_2}$$

$$\begin{aligned} & [\Omega]\Delta \Vdash^k [\Omega]\tau_1 : \kappa_1 \longrightarrow \kappa_2 \\ & \Delta \Vdash^k \tau_1 : \kappa'_1 \dashv \Theta_1 \wedge \Theta_1 \longrightarrow \Omega_1 \wedge \Omega \longrightarrow \Omega_1 \wedge [\Omega_1]\kappa'_1 = \kappa_1 \longrightarrow \kappa_2 \\ & \Delta \longrightarrow \Theta_1 \\ & \Delta \longrightarrow \Omega_1 \\ & [\Omega]\Delta \Vdash^k [\Omega]\tau_2 : \kappa_1 \\ & [\Omega]\tau_2 = [\Omega]([\Omega_1]\tau_2) = [\Omega_1]\tau_2 \\ & [\Omega]\Delta \Vdash^k [\Omega_1]\tau_2 : \kappa_1 \\ & [\Omega]\Delta \\ & = [\Omega]\Omega \\ & = [\Omega_1]\Omega_1 \\ & = [\Omega_1]\Theta_1 \\ & [\Omega_1]\Theta_1 \Vdash^k [\Omega_1]\tau_2 : \kappa_1 \\ & \Theta_1 \Vdash^k \tau_2 : \kappa'_2 \dashv \Theta_2 \wedge \Theta_2 \longrightarrow \Omega_2 \wedge \Omega_1 \longrightarrow \Omega_2 \wedge [\Omega_2]\kappa'_2 = \kappa_1 \\ & [\Omega_2]\kappa'_1 = [\Omega_2]([\Omega_1]\kappa'_1) = \kappa_1 \longrightarrow \kappa_2 \\ & \Theta_1 \Vdash^{kv} \kappa'_1 \\ & \Theta_2 \Vdash^{kv} \kappa'_2 \wedge \Theta_1 \longrightarrow \Theta_2 \\ & \Theta_2 \Vdash^{kv} \kappa'_1 \\ & \Theta_2 \Vdash^{kv} [\Theta_2]\kappa'_1 \wedge \Theta_2 \Vdash^{kv} [\Theta_2]\kappa'_2 \\ & \Theta_2 \Vdash^{kapp} [\Theta_2]\kappa'_1 \bullet [\Theta_2]\kappa'_2 : \kappa_3 \dashv \Theta \wedge \Theta \longrightarrow \Omega' \wedge \Omega_2 \longrightarrow \Omega' \wedge [\Omega']\kappa_3 = \kappa_2 \\ & \Delta \Vdash^k \tau_1 \tau_2 : \kappa_3 \dashv \Theta \\ & \Omega \longrightarrow \Omega' \end{aligned}$$

Given
I.H.
By Lemma [D.4](#)
By Lemma [D.20](#)
Given
By Lemma [D.18](#)
Follows directly

By Lemma [D.31](#)
By Lemma [D.34](#)
By Lemma [D.31](#)
Follows directly
I.H.
By Lemma [D.18](#)
By Lemma [D.6](#)
By Lemma [D.6](#)
By Lemma [D.10](#)
By Lemma [D.12](#)
By Lemma [D.47](#)
By rule [A-K-APP](#)
By Lemma [D.20](#)

□

Lemma D.49 (Completeness of Typing Data Constructor Declaration). *Given Δ ok and $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash_{\tau'}^{dc} \mathcal{D} \rightsquigarrow \tau$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash_{\tau'}^{dc} \mathcal{D} \rightsquigarrow \tau \dashv \Theta$.*

PROOF. Given

$$\frac{\text{DC-DECL} \quad \Sigma \vdash^k \overline{\tau}_i^i \rightarrow \tau : \star}{\Sigma \vdash_{\tau}^{\text{dc}} D \overline{\tau}_i^i \rightsquigarrow \overline{\tau}_i^i \rightarrow \tau}$$

Follows directly from Lemma D.48 and rule A-DC-DECL. \square

Lemma D.50 (Completeness of Typing Datatype Declaration). *Given Δ ok, and $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$ and $\Psi = [\Omega']\Gamma$.*

PROOF. We have

$$\frac{\text{DT-DECL} \quad (T : \overline{\kappa}_i^i \rightarrow \star) \in \Sigma \quad \overline{\Sigma, \overline{a}_i : \overline{\kappa}_i^i \vdash_{T \overline{a}_i}^{\text{dc}} \mathcal{D}_j \rightsquigarrow \tau_j}}{\Sigma \vdash^{\text{dt}} \text{data } T \overline{a}_i^i = \overline{\mathcal{D}_j^j} \rightsquigarrow D_j : \forall \overline{a}_i : \overline{\kappa}_i^i. \tau_j^j}$$

$(T : \overline{\kappa}_i^i \rightarrow \star) \in [\Omega]\Delta$ $(T : \kappa) \in \Delta \wedge [\Omega]\kappa = \overline{\kappa}_i^i \rightarrow \star$ $\Delta \longrightarrow \Omega$ $\Delta, \overline{\widehat{\alpha}}_i^i \longrightarrow \Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i$ $[\Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i] \kappa = \overline{\kappa}_i^i \rightarrow \star = [\Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i] (\overline{\widehat{\alpha}}_i^i \rightarrow \star)$ $[\Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i] \kappa$ $= [\Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i] ([\Delta, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i] \kappa)$ $= [\Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i] ([\Delta] \kappa)$ $\Delta, \overline{\widehat{\alpha}}_i \Vdash^{\mu} [\Delta] \kappa \approx \overline{\widehat{\alpha}}_i^i \rightarrow \star \dashv \Theta_1, \overline{\widehat{\alpha}}_i = \overline{\kappa}'_i^i$ $\wedge \Theta_1, \overline{\widehat{\alpha}}_i = \overline{\kappa}'_i^i \longrightarrow \Omega_1 \wedge \Omega, \overline{\widehat{\alpha}}_i = \overline{\kappa}_i^i \longrightarrow \Omega_1$ $[\Omega_1] \kappa_i = [\Omega_1] \kappa'_i = \kappa_i$ $\Delta, \overline{\widehat{\alpha}}_i \longrightarrow \Theta_1, \overline{\widehat{\alpha}}_i = \overline{\kappa}'_i^i$ $\Delta \longrightarrow \Theta_1$ $\Omega_1 = \Omega_{11}, \overline{\widehat{\alpha}}_1 = \kappa'', \Omega_{12} \wedge \Theta_1 \longrightarrow \Omega_{11}$ $\Omega \longrightarrow \Omega_{11}$ $[\Omega_{11}] \kappa'_i = [\Omega_{11}] \kappa'_i = \kappa_i$ $\Theta_1, \overline{a}_i : \overline{\kappa}'_i^i \longrightarrow \Omega_{11}, \overline{a}_i : \overline{\kappa}'_i^i$ $[\Omega_{11}, \overline{a}_i : \overline{\kappa}'_i^i] (\Theta_1, \overline{a}_i : \overline{\kappa}'_i^i)$ $= [\Omega_{11}] \Theta_1, \overline{a}_i : [\Omega_{11}] \overline{\kappa}'_i^i$ $= [\Omega_{11}] \Theta_1, \overline{a}_i : \overline{\kappa}_i^i$ $= [\Omega_{11}] \Omega_{11}, \overline{a}_i : \overline{\kappa}_i^i$ $= [\Omega] \Omega, \overline{a}_i : \overline{\kappa}_i^i$ $= [\Omega] \Delta, \overline{a}_i : \overline{\kappa}_i^i$ $[\Omega] \Delta, \overline{a}_i : \overline{\kappa}_i^i \vdash_{T \overline{a}_i}^{\text{dc}} \mathcal{D}_1 \rightsquigarrow \tau_1$ $[\Omega_{11}, \overline{a}_i : \overline{\kappa}'_i^i] (\Theta_1, \overline{a}_i : \overline{\kappa}'_i^i) \vdash_{T \overline{a}_i}^{\text{dc}} \mathcal{D}_1 \rightsquigarrow \tau_1$ $(\Theta_1, \overline{a}_i : \overline{\kappa}'_i^i) \Vdash_{T \overline{a}_i}^{\text{dc}} \mathcal{D}_1 \rightsquigarrow \tau_1 \dashv \Theta_2, \overline{a}_i : \overline{\kappa}'_i^i$	<p>Given</p> <p>By inversion</p> <p>Given</p> <p>By rule A-CTXE-SOLVE</p> <p>Follows directly</p> <p>By Lemma D.10</p> <p>$\widehat{\alpha}_i$ fresh</p> <p>By Lemma D.46 and inversion</p> <p>Above</p> <p>By Lemma D.17</p> <p>By Lemma D.4</p> <p>By Lemma D.17</p> <p>By Lemma D.17</p> <p>By Lemma D.17</p> <p>By Lemma D.19</p> <p>By rule A-CTXE-TVAR</p> <p>By definition</p> <p>By equations</p> <p>By Lemma D.31</p> <p>By Lemma D.34</p> <p>By Lemma D.31</p> <p>Given</p> <p>By equations</p> <p>By Lemma D.49 and inversion</p>
--	---

$\wedge \Theta_2, \overline{a_i : \kappa'_i}^i \longrightarrow \Omega_2 \wedge \Omega_{11}, \overline{a_i : \kappa'_i}^i \longrightarrow \Omega_2$	By Lemma D.49
$\Omega_2 = \Omega_{21}, a_1 : \kappa'_1, \Omega_{22} \wedge \Omega_{11} \longrightarrow \Omega_{21}$	By Lemma D.17
$\Theta_2 \longrightarrow \Omega_{21}$	By Lemma D.17
$\Theta_2, \overline{a_i : \kappa'_i}^i \longrightarrow \Omega_{21}, \overline{a_i : \kappa'_i}^i$	By rule A-CTXE-TVAR
$\Omega_{11} \longrightarrow \Omega_{21}$	By Lemma D.17
$\Omega_{11}, \overline{a_i : \kappa'_i}^i \longrightarrow \Omega_{21}, \overline{a_i : \kappa'_i}^i$	By rule A-CTXE-TVAR

We repeat the process for each j . Let $\Theta_{n+1}, \overline{a_i : \kappa'_i}^i$ and $\Omega_{n+1}, \overline{a_i : \kappa'_i}^i$ be the final output context and the complete context. And $\Theta_{n+1}, \overline{a_i : \kappa'_i}^i \longrightarrow \Omega_{n+1}, \overline{a_i : \kappa'_i}^i$.

By Lemma D.20 we have $\Omega \longrightarrow \Omega_{n+1}$.

By Lemma D.29 we have $[\Omega_{n+1}] \kappa'_i = [\Omega_{11}] \kappa'_i = \kappa_i$.

So collecting all the hypothesis, by rule A-DT-DECL we get $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \vdash \Theta$. And $[\Omega_{n+1}] \Gamma = \Psi$. Let $\Omega' = \Omega_{n+1}$.

□

Theorem D.51 (Completeness of Typing a Group). *Given Ω ok, if $[\Omega] \Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i ; \overline{\Psi}_i^i$, then there exists $\overline{\kappa}_i^i, \overline{\Gamma}_i^i, \Theta$, and Ω' , such that $\Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i ; \overline{\Gamma}_i^i \vdash \Theta$, where $\Theta \longrightarrow \Omega'$, and $[\Omega'] \kappa'_i = \kappa_i$, and $\overline{\Psi}_i = [\Omega'] \overline{\Gamma}_i^i$.*

PROOF. We have

$\Theta_1 = \Omega, \overline{\widehat{\alpha}_i}^{i \in 1..n}, \overline{T_i : \widehat{\alpha}_i}^{i \in 1..n}$	Let
$\Omega_1 = \Omega, \overline{\widehat{\alpha}_i = \kappa_i}^{i \in 1..n}, \overline{T_i : \widehat{\alpha}_i}^{i \in 1..n}$	Let
$\Theta_1 \longrightarrow \Omega_1$	By Lemma D.21
$[\Omega] \Omega, \overline{T_i : \kappa_i}^i \Vdash^{\text{dt}} \mathcal{T}_1 \rightsquigarrow \Psi_1$	Given
$[\Omega_1] \Theta_1 \Vdash^{\text{dt}} \mathcal{T}_1 \rightsquigarrow \Psi_1$	By definition
$\Theta_1 \Vdash^{\text{dt}} \mathcal{T}_1 \rightsquigarrow \Gamma_1 \vdash \Theta_2 \wedge \Theta_2 \longrightarrow \Omega_2 \wedge \Omega_1 \longrightarrow \Omega_2 \wedge \Psi_1 = [\Omega_2] \Gamma_1$	(1) By Lemma D.50
$[\Omega] \Omega, \overline{T_i : \kappa_i}^i \Vdash^{\text{dt}} \mathcal{T}_2 \rightsquigarrow \Psi_2$	Given
$[\Omega_2] \Theta_2$	By Lemma D.31
$= [\Omega_2] \Omega_2$	By Lemma D.34
$= [\Omega_1] \Omega_1$	By definition
$= [\Omega] \Omega, \overline{T_i : \kappa_i}^i$	Substitute the equation
$[\Omega_2] \Theta_2 \Vdash^{\text{dt}} \mathcal{T}_2 \rightsquigarrow \Psi_2$	(2) By Lemma D.50
$\Theta_2 \Vdash^{\text{dt}} \mathcal{T}_2 \rightsquigarrow \Gamma_2 \vdash \Theta_3 \wedge \Theta_3 \longrightarrow \Omega_3 \wedge \Omega_2 \longrightarrow \Omega_3 \wedge \Psi_2 = [\Omega_3] \Gamma_2$	
By repeating the process from (1) to (2) for each i , we can get	
$\overline{\Theta_i} \Vdash^{\text{dt}} \mathcal{T}_i \rightsquigarrow \overline{\Gamma_i} \vdash \overline{\Theta_{i+1}} \wedge \overline{\Theta_{i+1}} \longrightarrow \overline{\Omega_{i+1}} \wedge \overline{\Psi_i} = [\overline{\Omega_{i+1}}] \overline{\Gamma_i}^i$	
$\Omega' = \overline{\Omega_{i+1}}$	Let
$\overline{[\Omega'] \widehat{\alpha}_i = [\Omega'] ([\Omega_1] \widehat{\alpha}_i)}$	By Lemma D.18
$\overline{[\Omega'] \widehat{\alpha}_i = \kappa_i}^i$	Namely
$\overline{[\Omega'] \Gamma_i = [\Omega_{i+1}] \Gamma_i}^i$	By Lemma D.30
$\overline{[\Omega'] \Gamma_i = \Psi_i}^i$	Namely

□

E PROOF FOR HASKELL98 WITH KIND PARAMETERS

E.1 List of Lemmas

Theorem E.1 (Principality of Haskell98 with Kind Parameters). *If $\Sigma \Vdash^{\text{gfp}} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i; \overline{\Psi}_i^i$, then there exists some $\overline{\kappa}'_i^i$ such that $\Sigma \vdash \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow^{\text{p}} \overline{\kappa}'_i^i$.*

Theorem E.2 (Completeness of Typing Programs with Kind Parameters). *Given algorithmic contexts Ω, Γ , and a program pgm , if $[\Omega]\Omega; [\Omega]\Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$, then $\Omega; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$.*

E.2 Proofs

Theorem E.1 (Principality of Haskell98 with Kind Parameters). *If $\Sigma \Vdash^{\text{gfp}} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i; \overline{\Psi}_i^i$, then there exists some $\overline{\kappa}'_i^i$ such that $\Sigma \vdash \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow^{\text{p}} \overline{\kappa}'_i^i$.*

PROOF. We have

$$\begin{array}{l|l} \Sigma \Vdash^{\text{gfp}} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i; \overline{\Psi}_i^i & \text{Given} \\ \Omega = \Sigma & \text{Let} \\ \Omega \Vdash^{\text{gfp}} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}'_i^i; \overline{\Gamma}_i^i \dashv \Theta & \text{By Theorem D.51} \\ \Theta \longrightarrow \Omega' \wedge [\Omega']\kappa'_i = \kappa_i \wedge [\Omega']\Gamma_i = \Psi_i & \text{Above} \end{array}$$

We solve all unsolved kind unification variables in Θ with fresh kind parameters to get Ω_1 . Then we choose $\overline{\kappa}''_i = [\Omega_1]\kappa'_i$, and we prove $\Sigma \vdash \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow^{\text{p}} \overline{\kappa}''_i$.

$$\begin{array}{l|l} \Theta \longrightarrow \Omega_1 & \text{By Lemma D.21} \\ \Omega, \overline{\widehat{\alpha}}_i^i, \overline{T}_i : \overline{\widehat{\alpha}}_i^i \longrightarrow \Omega_1 & \text{By Lemma D.20} \\ \Omega_1 = \Omega_{11}, \widehat{\alpha}_1 = \kappa_{11}, \Omega_{12}, \overline{T}_i : \overline{\widehat{\alpha}}_i^i, \overline{\Omega}'_i & \text{By Lemma D.17} \\ \wedge \Omega \longrightarrow \Omega_{11} \wedge \Omega_{12} \mathbf{soft} \wedge \overline{\Omega}'_i \mathbf{soft} & \text{Above} \\ [\Omega_1]\Theta & \\ = [\Omega_1](\Omega, \overline{\widehat{\alpha}}_i^i, \overline{T}_i : \overline{\widehat{\alpha}}_i^i) & \text{By Lemma D.33} \\ = [\Omega]\Omega, \overline{T}_i : [\Omega_1]\overline{\widehat{\alpha}}_i^i & \text{By definition and Lemma D.32} \\ = \Sigma, \overline{T}_i : \overline{\kappa}''_i^i & \text{By Lemma D.17} \\ \Sigma \Vdash^{\text{gfp}} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}''_i^i; \overline{\Psi}_i^i & \text{repeat Lemma D.43} \end{array}$$

For any $\overline{\kappa}'_i^i$ such that $\Sigma \Vdash^{\text{gfp}} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}'_i^i; \overline{\Psi}_i^i$, by Theorem D.51 we know there exists some Ω' such that $\Theta \longrightarrow \Omega'$ and $[\Omega']\kappa'_i = \kappa_i$ and $[\Omega']\Gamma_i = \Psi_i$. Now we construct a kind parameter substitution S . If in Θ , we have an unsolved kind unification variable $\widehat{\alpha}$, which maps to a parameter P in Ω_1 . then S maps P to $[\Omega']\widehat{\alpha}$. Because $\Theta \longrightarrow \Omega'$, then $S(\Omega_1) \longrightarrow \Omega'$ by Lemma D.25. So $S(\kappa'_i) = S([\Omega_1]\kappa'_i) = [S(\Omega_1)]\kappa'_i$. By Lemma D.29, we have $[S(\Omega_1)]\kappa'_i = [\Omega']\kappa'_i = \kappa_i$. Similarly we have $S(\Psi'_i) = \Psi_i$. □

Theorem E.2 (Completeness of Typing Programs with Kind Parameters). *Given algorithmic contexts Ω, Γ , and a program pgm , if $[\Omega]\Omega; [\Omega]\Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$, then $\Omega; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \sigma$.*

PROOF. By induction on typing programs.

- Case

$$\frac{\text{PGM-EXPR}}{\Sigma; \Psi \vdash e : \sigma} \\ \Sigma; \Psi \Vdash^{\text{pgm}} e : \sigma$$

Follows trivially by rule **A-PGM-EXPR**.

- Case

PGM-DTP

$$\frac{\Sigma \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i; \overline{\Psi}_i^i \quad \Sigma \vdash \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow^p \overline{\kappa}_i^i \quad \Sigma, \overline{T}_i : S^*(\kappa_i)^i; \Psi, S^*(\Psi_i)^i \Vdash^{\text{pgm}} \text{pgm} : \sigma}{\Sigma; \Psi \Vdash^{\text{pgm}} \text{rec } \overline{\mathcal{T}}_i^i; \text{pgm} : \sigma}$$

$$\begin{array}{l|l} \Sigma \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}_i^i; \overline{\Psi}_i^i & \text{Given} \\ \Sigma \vdash \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow^p \overline{\kappa}_i^i & \text{Given} \\ \Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\kappa}'_i^i; \overline{\Gamma}_i^i \dashv \Theta & \text{By Theorem D.51} \\ \Theta \longrightarrow \Omega' & \text{Above} \\ \frac{[\Omega']\kappa'_i = \kappa_i}{[\Omega']\Gamma_i = \Psi_i} & \text{Above} \end{array}$$

Because from Theorem E.1 we know that if we solve all unsolved kind unification variables in Θ with fresh parameters to get Ω_1 , then $[\Omega_1]\kappa'_i$ are principal kinds. Because $\overline{\kappa}_i^i$ are principal kinds, then $[\Omega_1]\kappa'_i$ and $\overline{\kappa}_i^i$ are equivalent up to renaming of type parameters. Suppose $\Theta \longrightarrow \Omega_2$, then $[\Omega_2]\kappa'_i = S^*(\kappa_i)^i$. Similarly we can prove $[\Omega_2]\Gamma_i = S^*(\Psi_i)^i$.

$$\begin{array}{l|l} \Omega, \overline{\alpha}_i^{i \in 1..n}, \overline{T}_i : \overline{\alpha}_i^{i \in 1..n} \longrightarrow \Theta & \text{By Lemma D.8} \\ \Theta \longrightarrow \Omega_2 & \text{By Lemma D.13} \\ \Omega, \overline{\alpha}_i^{i \in 1..n}, \overline{T}_i : \overline{\alpha}_i^{i \in 1..n} \longrightarrow \Omega_2 & \text{By Lemma D.20} \\ \Omega, \overline{\alpha}_i = [\Omega_2]\kappa'_i^{i \in 1..n}, \overline{T}_i : \overline{\alpha}_i^{i \in 1..n} \longrightarrow \Omega_2 & \text{By Lemma D.25} \\ [\Omega_2]\Omega_2 & \\ = [\Omega, \overline{\alpha}_i = [\Omega_2]\kappa'_i^{i \in 1..n}, \overline{T}_i : \overline{\alpha}_i^{i \in 1..n}] & \\ (\Omega, \overline{\alpha}_i = [\Omega_2]\kappa'_i^{i \in 1..n}, \overline{T}_i : \overline{\alpha}_i^{i \in 1..n}) & \text{By Lemma D.34} \\ = [\Omega]\Omega, \overline{T}_i : [\Omega]([\Omega_2]\kappa'_i^{i \in 1..n}) & \text{By definition} \\ = [\Omega]\Omega, \overline{T}_i : S^*(\kappa_i)^i & \text{By substituting the equation} \\ [\Omega_2](\Gamma, \overline{\Gamma}_i^i) & \\ = [\Omega_2]\Gamma, [\Omega_2]\overline{\Gamma}_i^i & \text{By definition} \\ = [\Omega]\Gamma, S^*(\Psi_i)^i & \text{By Lemma D.30 and Lemma D.19} \\ [\Omega]\Omega, \overline{T}_i : S^*(\kappa_i)^i; [\Omega]\Gamma, S^*(\Psi_i)^i \vdash_p \text{pgm} : \sigma & \text{Given} \\ [\Omega_2]\Omega_2; [\Omega_2](\Gamma, \overline{\Gamma}_i^i) \Vdash^{\text{pgm}} \text{pgm} : \sigma & \text{By substituting the equations} \\ \Omega_2; \Gamma, \overline{\Gamma}_i^i \Vdash^{\text{pgm}} \text{pgm} : \sigma & \text{I.H.} \\ \Omega; \Gamma \Vdash^{\text{pgm}} \text{rec } \overline{\mathcal{T}}_i^i; \text{pgm} : \sigma & \text{By rule A-PGM-DT} \end{array}$$

□

F PROOF FOR POLYKINDS

F.1 List of Lemmas

F.1.1 Well-formedness of Declarative Type System.

Lemma F.1 (Well-formedness of Declarative Instantiation). *If $\Sigma \Vdash^{\text{ela}} \mu_1 : \eta_1$, and $\Sigma \Vdash^{\text{inst}} \mu_1 : \eta_1 \sqsubseteq \eta_2 \rightsquigarrow \mu_2$, then $\Sigma \Vdash^{\text{ela}} \mu_2 : \eta_2$.*

Lemma F.2 (Well-formedness of Declarative Kinding). *We have:*

- if $\Sigma \Vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu$, then $\Sigma \Vdash^{\text{ela}} \mu : \eta$;
- if $\Sigma \Vdash^{\text{k}} \sigma \Leftarrow \eta \rightsquigarrow \mu$, then $\Sigma \Vdash^{\text{ela}} \mu : \eta$.

Lemma F.3 (Well-formedness of Declarative Elaborated Kinding). *If Σok , and $\Sigma \Vdash^{\text{ela}} \mu : \eta$, then $\Sigma \Vdash^{\text{ela}} \eta : \star$.*

Lemma F.4 (Well-formedness of Declarative Typing Signature). *If Σok , and $\Sigma \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$, then $\Sigma \Vdash^{\text{ela}} \eta : \star$.*

Lemma F.5 (Well-formedness of Declarative Typing Data Constructor Declaration). *If Σok , and $\Sigma \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu$, then $\Sigma \Vdash^{\text{ela}} \mu : \star$.*

Lemma F.6 (Well-formedness of Declarative Typing Datatype Declaration). *If Σok , and $\Sigma \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, then $\Sigma \vdash \Psi$.*

Lemma F.7 (Well-formedness of Declarative Generalization). *If Σok , and $\Sigma \vdash \Psi_1$ and $\Sigma \Vdash_{\phi^c}^{\text{gen}} \Psi_1 \rightsquigarrow \Psi_2$, then $\Sigma \vdash \Psi_2$.*

F.1.2 Well-formedness of Algorithmic Type System.

Lemma F.8 (Well-formedness of Promotion). *If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \text{ok}$, and $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash^{\text{ela}} \rho_1 : \omega_2$, and $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash_{\alpha}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \vdash \Theta$, then $\Theta = \Theta_1, \widehat{\alpha} : \omega, \Theta_2$, and $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta$, and $\Theta_1 \Vdash^{\text{ela}} \rho_2 : [\Theta]\omega_2$, and Θok . By weakening, there is also $\Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\omega_2$. Similar lemma holds when in the input context, $\widehat{\alpha} : \omega$ is in a local scope.*

Lemma F.9 (Well-formedness of Moving). *If $\Delta_1 \dashv\vdash^{\text{mv}} \Delta_2 \rightsquigarrow \Theta$, then $\text{topo}(\Delta_1, \Delta_2) = \Theta$.*

Lemma F.10 (Well-formedness of Unification). *If Δok , and $\Delta \Vdash^{\text{u}} \kappa_1 \approx \kappa_2 \vdash \Theta$, then $\Delta \longrightarrow \Theta$, and Θok .*

Lemma F.11 (Well-formedness of Instantiation). *If $\Delta \Vdash^{\text{inst}} \rho_1 : \eta_1 \sqsubseteq \eta_2 \rightsquigarrow \rho_2 \vdash \Theta$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \eta_1$, then $\Delta \longrightarrow \Theta$, and Θok , and $\Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\eta_2$.*

Lemma F.12 (Well-formedness of Quantification Check). *If $\Delta_1, a : \omega, \Delta_2 \text{ok}$, and $\Delta_2 \hookrightarrow a$, then $\Delta_1, \Delta_2 \text{ok}$.*

Lemma F.13 (Well-formedness of Unsolved). *If $\Delta_1, \Delta_2 \text{ok}$, and $\Delta_2 \text{soft}$, then $\Delta_1, \text{unsolved}(\Delta_2) \text{ok}$.*

Lemma F.14 (Well-formedness of topo). *If $\Delta_1, \Delta_2 \text{ok}$, then $\Delta_1, \text{topo}(\Delta_2) \text{ok}$.*

Lemma F.15 (Well-formedness of Kinding). *Given Δok ,*

- if $\Delta \Vdash^k \sigma : \eta \rightsquigarrow \mu \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok and $\Theta \Vdash^{\text{ela}} \mu : [\Theta]\eta$;
- if $\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \eta \rightsquigarrow \mu \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok and $\Theta \Vdash^{\text{ela}} \mu : [\Theta]\eta$.
- if $\Delta \Vdash^{\text{kapp}} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 \dashv \Theta$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \eta$, then $\Delta \longrightarrow \Theta$, and Θ ok, and $\Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\omega$.

Lemma F.16 (Well-formedness of Elaborated Kinding). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu : \eta$, then $\Delta \Vdash^{\text{ela}} \eta : \star$, and $[\Delta]\eta = \eta$.*

Lemma F.17 (Well-formedness of Typing Signature). *If Ω ok, and $\Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \mu$, then $\Omega \Vdash^{\text{ela}} \mu : \star$.*

Lemma F.18 (Well-formedness of Typing Data Constructor Declaration). *If Δ ok, and $\Delta \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \Vdash^{\text{ela}} \mu : \star$.*

Lemma F.19 (Well-formedness of Typing Datatype Declaration). *If Δ ok, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \Vdash^{\text{ectx}} \Gamma$.*

Lemma F.20 (Well-formedness of Generalization). *If Δ ok, and $\Delta \Vdash^{\text{ectx}} \Gamma_1$, and $\Delta \Vdash_{\phi^c}^{\text{gen}} \Gamma_1 \rightsquigarrow \Gamma_2$, then $\Delta \Vdash^{\text{ectx}} \Gamma_2$.*

F.1.3 Properties of Context Extension.

Lemma F.21 (Declaration Preservation). *If $\Delta \longrightarrow \Theta$, if a type constructor or a type variable or a kind unification variable is declared in Δ , then it is declared in Θ .*

Lemma F.22 (Extension Weakening). *Given $\Delta \longrightarrow \Theta$, if $\Delta \Vdash^{\text{ela}} \mu : \eta$, then $\Theta \Vdash^{\text{ela}} \mu : [\Theta]\eta$.*

Definition F.23 (Contextual Size).

$ \Delta \vdash \star $	$= 1$
$ \Delta \vdash a $	$= 1$
$ \Delta \vdash \text{Int} $	$= 1$
$ \Delta \vdash T $	$= 1$
$ \Delta \vdash \rightarrow $	$= 1$
$ \Delta \vdash \omega_1 \omega_2 $	$= 1 + \Delta \vdash \omega_1 + \Delta \vdash \omega_2 $
$ \Delta \vdash \omega_1 @ \omega_2 $	$= 1 + \Delta \vdash \omega_1 + \Delta \vdash \omega_2 $
$ \Delta[\widehat{\alpha} : \omega] \vdash \widehat{\alpha} $	$= 1$
$ \Delta[\widehat{\alpha} : \omega = \rho] \vdash \widehat{\alpha} $	$= 1 + \Delta[\widehat{\alpha} : \omega = \rho] \vdash \omega $
$ \Delta \vdash \forall a : \rho. \omega $	$= 1 + \Delta \vdash \rho + \Delta \vdash \omega $
$ \Delta \vdash \forall \{a : \rho\}. \omega $	$= 1 + \Delta \vdash \rho + \Delta \vdash \omega $

Lemma F.24 (Substitution Kinding). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu : \eta$, then $\Delta \Vdash^{\text{ela}} [\Delta]\mu : \eta$.*

Lemma F.25 (Soft Substitution Kinding). *If Δ_1, Δ_2 ok, and Δ_2 soft, and $\Delta_1, \Delta_2 \Vdash^{\text{ela}} \mu : \eta$, then $\Delta_1, \text{unsolved}(\Delta_2) \Vdash^{\text{ela}} [\Delta_2]\mu : \eta$.*

Lemma F.26 (Reflexivity of Context Extension). *If Δ ok, then $\Delta \longrightarrow \Delta$.*

Lemma F.27 (Well-formedness of Context Extension). *If Δ ok, and $\Delta \longrightarrow \Theta$, then Θ ok.*

Definition F.28 (Softness). *A context Δ is soft iff it contains only of $\widehat{\alpha}$ and $\widehat{\alpha} = \kappa$ declarations, including local scopes.*

Lemma F.29 (Extension Order).

- (1) *If $\Delta_1, a : \omega, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, a : \omega, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.*
- (2) *If $\Delta_1, T : \eta, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, T : \eta, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.*
- (3) *If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \Theta', \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$, and Θ' is either $\widehat{\alpha} : \omega$ or $\widehat{\alpha} : \omega = \rho$ for some ρ . Moreover, if Δ_2 **soft**, then Θ_2 **soft**.*
- (4) *If $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$, and $[\Theta_1]\rho_1 = [\Theta_1]\rho_2$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.*
- (5) *If $\Delta_1, \{\Delta\}, \Delta_2 \longrightarrow \Theta$, then $\Theta = \Theta_1, \{\Theta\}, \Theta_2$, where $\Delta_1 \longrightarrow \Theta_1$. Moreover, if Δ_2 **soft**, then Θ_2 **soft**.*

Lemma F.30 (Substitution Extension Invariance). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu : \eta$, and $\Delta \longrightarrow \Theta$, then $[\Theta]\kappa = [\Theta]([\Delta]\mu)$ and $[\Theta]\kappa = [\Delta]([\Theta]\mu)$. As a corollary, if $\Delta \Vdash^{\text{ela}} \mu_1 : \eta_1$, $\Delta \Vdash^{\text{ela}} \mu_2 : \eta_2$, and $[\Delta]\mu_1 = [\Delta]\mu_2$, then $[\Theta]\mu_1 = [\Theta]\mu_2$.*

Lemma F.31 (Substitution Stability). *If Δ_1, Δ_2 ok, and $\Delta_1 \Vdash^{\text{ela}} \rho : \omega$, then $[\Delta_1]\rho = [\Delta_1, \Delta_2]\rho$.*

Lemma F.32 (Transitivity of Context Extension). *If Δ' ok, and $\Delta' \longrightarrow \Delta$, and $\Delta \longrightarrow \Theta$, then $\Delta' \longrightarrow \Theta$.*

Lemma F.33 (Solution Admissibility for Extension).

- *If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2$ ok and $\Delta_1 \Vdash^{\text{ela}} \rho : [\Delta_1]\omega$, then $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} : \omega = \rho, \Delta_2$.*
- *If $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2$ ok and $\Delta_1, \Delta_3 \Vdash^{\text{ela}} \rho : [\Delta_1, \Delta_3]\omega$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho, \Delta_4\}, \Delta_2$.*

Lemma F.34 (Solved Variable Addition for Extension).

- *If Δ_1, Δ_2 ok and $\Delta_1 \Vdash^{\text{ela}} \rho : [\Delta_1]\omega$, then $\Delta_1, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} : \omega = \rho, \Delta_2$.*
- *If $\Delta_1, \{\Delta_2, \Delta_3\}, \Delta_4$ ok and $\Delta_1, \Delta_2 \Vdash^{\text{ela}} \rho : [\Delta_1, \Delta_2]\omega$, then $\Delta_1, \{\Delta_2, \Delta_3\}, \Delta_4 \longrightarrow \Delta_1, \{\Delta_2, \widehat{\alpha} : \omega = \rho, \Delta_3\}, \Delta_4$.*

Lemma F.35 (Unsolved Variable Addition).

- *If Δ_1, Δ_2 ok and $\Delta_1 \Vdash^{\text{ela}} \omega : \star$ then $\Delta_1, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} : \omega, \Delta_2$.*
- *If $\Delta_1, \{\Delta_2, \Delta_3\}, \Delta_4$ ok and $\Delta_1, \Delta_2 \Vdash^{\text{ela}} \omega : \star$, then $\Delta_1, \{\Delta_2, \Delta_3\}, \Delta_4 \longrightarrow \Delta_1, \{\Delta_2, \widehat{\alpha} : \omega, \Delta_3\}, \Delta_4$.*

Lemma F.36 (Parallel Admissibility).

- *If $\Delta_1 \longrightarrow \Theta_1$, and Δ_1, Δ_2 ok, and $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$, and Δ_2 is fresh w.r.t. Θ_1 , then:

 - *if $\Delta_1 \Vdash^{\text{ela}} \omega : \star$, then $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega, \Theta_2$;*
 - *if $\Theta_1 \Vdash^{\text{ela}} \rho : [\Theta_1]\omega$, then $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho, \Theta_2$;*
 - *if $[\Theta_1]\rho_1 = [\Theta_1]\rho_2$, then $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_2$.**
- *If $\Delta_1, \{\Delta_3\} \longrightarrow \Theta_1, \{\Theta_3\}$, and $\Delta_1, \{\Delta_3, \Delta_4\}, \Delta_2$ ok, and $\Delta_1, \{\Delta_3, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \Theta_4\}, \Theta_2$, and Δ_2, Δ_4 is fresh w.r.t. Θ_1, Θ_3 , then:

 - *if $\Delta_1, \{\Delta_3\} \Vdash^{\text{ela}} \omega : \star$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega, \Theta_4\}, \Theta_2$;*
 - *if $\Theta_1, \{\Theta_3\} \Vdash^{\text{ela}} \rho : [\Theta_1, \Theta_3]\omega$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho, \Theta_4\}, \Theta_2$;**

- if $[\Theta_1, \Theta_3]\rho_1 = [\Theta_1, \Theta_3]\rho_2$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$.

Lemma F.37 (Parallel Extension Solution).

- If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_2$, and $[\Theta_1]\rho_1 = [\Theta_1]\rho_2$, then $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \kappa_2, \Theta_2$.
- If $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$, and $[\Theta_1, \Theta_3]\rho_1 = [\Theta_1, \Theta_3]\rho_2$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$.

Lemma F.38 (Parallel Variable Update).

- If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho, \Theta_2$, and $\Delta_1 \Vdash^{\text{ela}} \rho_1 : [\Delta_1]\omega$, and $\Theta_1 \Vdash^{\text{ela}} \rho_2 : [\Theta_1]\omega$, and $[\Theta_1]\rho = [\Theta_1]\rho_1 = [\Theta_1]\rho_2$, then $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_2$.
- If $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho, \Theta_4\}, \Theta_2$, and $\Delta_1, \Delta_3 \Vdash^{\text{ela}} \rho_1 : [\Delta_1, \Delta_3]\omega$, and $\Theta_1, \Theta_3 \Vdash^{\text{ela}} \rho_2 : [\Theta_1, \Theta_3]\omega$, and $[\Theta_1]\rho = [\Theta_1]\rho_1 = [\Theta_1]\rho_2$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$.

F.1.4 Properties of Complete Context.

Lemma F.39 (Type Constructor Preservation). If Δ ok, then $(T : \eta) \in \Delta$, and $\Delta \longrightarrow \Omega$, then $(T : [\Omega]\eta) \in [\Omega]\Delta$.

Lemma F.40 (Type Variable Preservation). If $(a : \omega) \in \Delta$, and $\Delta \longrightarrow \Omega$, then $(a : [\Omega]\omega) \in [\Omega]\Delta$.

Lemma F.41 (Finishing Kinding). If Ω ok, and $\Omega \Vdash^{\text{ela}} \rho : \omega$, and $\Omega \longrightarrow \Omega'$, then $[\Omega]\rho = [\Omega']\rho$.

Lemma F.42 (Finishing Term Contexts). If Ω ok, and $\Omega \Vdash^{\text{ectx}} \Gamma$, and $\Omega \longrightarrow \Omega'$, then $[\Omega']\Gamma = [\Omega]\Gamma$.

Lemma F.43 (Stability of Complete Contexts). If $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta = [\Omega]\Omega$.

Lemma F.44 (Softness Goes Away). If $\Delta_1, \Delta_2 \longrightarrow \Omega_1, \Omega_2$ where $\Delta_1 \longrightarrow \Omega_1$, and Δ_2 soft, then $[\Omega_1, \Omega_2](\Delta_1, \Delta_2) = [\Omega_1]\Delta_1$.

Lemma F.45 (Confluence of Completeness). If $\Delta_1 \longrightarrow \Omega$, and $\Delta_2 \longrightarrow \Omega$, then $[\Omega]\Delta_1 = [\Omega]\Delta_2$.

Lemma F.46 (Finishing Completions). If Ω ok, and $\Omega \longrightarrow \Omega'$, then $[\Omega']\Omega' = \text{topo}([\Omega]\Omega)$.

F.1.5 Termination.

Lemma F.47 (Promotion Preserves $\langle \Delta \rangle$). If $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$, then $\langle \Delta \rangle = \langle \Theta \rangle$.

Lemma F.48 (Unification Makes Progress). If $\Delta \Vdash^{\mu} \omega_1 \approx \omega_2 \dashv \Theta$, then either $\Theta = \Delta$, or $\langle \Theta \rangle < \langle \Delta \rangle$.

Lemma F.49 (Promotion Preserves $|\rho|$). Given a context $\Delta[\widehat{\alpha}]$ ok, if $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$, then for all ρ , we have $|\Delta[\widehat{\alpha}]\rho| = |[\Theta]\rho|$.

Theorem F.50 (Promotion Terminates). Given a context $\Delta[\widehat{\alpha}]$ ok, and a kind ρ_1 with $[\Delta]\rho_1 = \rho_1$, it is decidable whether there exists Θ such that $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$.

Theorem F.51 (Unification Terminates). Given a context Δ ok, and kinds ρ_1 and ρ_2 , where $[\Delta]\rho_1 = \rho_1$, and $[\Delta]\rho_2 = \rho_2$, it is decidable whether there exists Θ such that $\Delta \Vdash^{\mu} \rho_1 \approx \rho_2 \dashv \Theta$.

F.1.6 Source of Unification Variables.

Lemma F.52 (Source of Unification Variables). *If $\Delta \Vdash^k \sigma : \eta \rightsquigarrow \mu \dashv \Theta$, then for any $\widehat{\alpha} \in \text{unsolved}(\Theta)$, either $\widehat{\alpha} \in \text{fkv}([\Theta]\mu)$, or there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$ such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$.*

F.1.7 Soundness of Algorithm.

Lemma F.53 (Soundness of Promotion). *If Δ ok, and $[\Delta]\omega_1 = \omega_1$, and $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$, then $[\Theta]\omega_1 = [\Theta]\omega_2 = \omega_2$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\omega_1 = [\Omega]\omega_2$.*

Lemma F.54 (Soundness of Unification). *If Δ ok, and $\Delta \Vdash^{\mu} \omega_1 \approx \omega_2 \dashv \Theta$, then $[\Theta]\omega_1 = [\Theta]\omega_2$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\omega_1 = [\Omega]\omega_2$.*

Lemma F.55 (Soundness of Instantiation). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu_1 : \eta$, and $\Delta \Vdash^{\text{ela}} \omega : \star$, and $\Delta \Vdash^{\text{inst}} \mu_1 : \eta \sqsubseteq \omega \rightsquigarrow \mu_2 \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\mu_1 : [\Omega]\eta \sqsubseteq [\Omega]\omega \rightsquigarrow [\Omega]\mu_2$.*

Lemma F.56 (Soundness of Kinding). *If Δ ok, we have*

- if $\Delta \Vdash^k \sigma : \eta \rightsquigarrow \mu \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^k \sigma : [\Omega]\eta \rightsquigarrow [\Omega]\mu$;
- if $\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \eta \rightsquigarrow \mu \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{kc}} \sigma \Leftarrow [\Omega]\eta \rightsquigarrow [\Omega]\mu$.
- if $\Delta \Vdash^{\text{kapp}} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 \dashv \Theta$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \eta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\rho_1 : [\Omega]\eta \sqsubseteq (\omega_1 \rightarrow [\Omega]\omega) \rightsquigarrow \rho_3$, and $[\Omega]\Delta \Vdash^{\text{kc}} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_4$. and $[\Omega]\rho_2 = \rho_3 \rho_4$.

Lemma F.57 (Soundness of Elaborated Kinding). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu : \eta$, and $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{ela}} [\Omega]\mu : [\Omega]\eta$.*

Lemma F.58 (Soundness of Typing Signature). *If Δ ok, and $\Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$, then $[\Omega]\Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$.*

Lemma F.59 (Soundness of Typing Data Constructor Decl.). *If Δ ok, and $\Delta \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash_{([\Omega]\rho)}^{\text{dc}} \mathcal{D} \rightsquigarrow [\Omega]\mu$.*

Lemma F.60 (Soundness of Typing Datatype Decl.). *If Δ ok, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow [\Omega]\Gamma$.*

Lemma F.61 (Soundness of Typing Program). *If $\Omega; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \mu$, then $[\Omega]\Omega; [\Omega]\Gamma \Vdash^{\text{pgm}} \text{pgm} : [\Omega]\mu$.*

F.1.8 Principality.

Lemma F.62 (Completeness of Promotion). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\widehat{\alpha} \in \Delta$, and $\Delta \Vdash^{\text{ela}} \rho : \omega$, and $[\Delta]\widehat{\alpha} = \widehat{\alpha}$, and $[\Delta]\rho = \rho$, if ρ does not depend on $\widehat{\alpha}$ in the dependency graph, then there exists ρ_2 , Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$, and $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho \rightsquigarrow \rho_2 \dashv \Theta$.*

Lemma F.63 (Completeness of Unification). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \omega$ and $\Delta \Vdash^{\text{ela}} \rho_2 : \omega$, and $[\Delta]\rho_1 = \rho_1$ and $[\Delta]\rho_2 = \rho_2$, if $[\Omega]\rho_1 = [\Omega]\rho_2$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\mu} \rho_1 \approx \rho_2 \dashv \Theta$.*

Lemma F.64 (Completeness of Instantiation). *Given $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{ela}} \rho : \eta$ and $\Delta \Vdash^{\text{ela}} \omega : \star$, and $[\Delta]\eta = \eta$ and $[\Delta]\omega = \omega$, if $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\rho_1 : [\Omega]\eta \sqsubseteq [\Omega]\omega \rightsquigarrow \rho_2$, then there exists ρ'_2 , Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\text{inst}} \rho_1 : \eta \sqsubseteq \omega \rightsquigarrow \rho'_2 \dashv \Theta$, and $[\Omega']\rho'_2 = \rho_2$.*

Lemma F.65 (Principality of Kinding).

- Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash^k \sigma : \eta \rightsquigarrow \mu$, and $\Delta \Vdash^k \sigma : \eta' \rightsquigarrow \mu' \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$. Moreover, $[\Omega']\eta' = \eta$. Furthermore, if μ and μ' are monotypes, then $[\Omega']\mu' = \mu$.
- Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash^{kc} \sigma \Leftarrow [\Omega]\eta \rightsquigarrow \mu$, and $\Delta \Vdash^{kc} \sigma \Leftarrow \eta \rightsquigarrow \mu' \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$. Furthermore, if μ and μ' are monotypes, then $[\Omega']\mu' = \mu$.
- Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash^{inst} [\Omega]\rho_1 : [\Omega]\eta \sqsubseteq (\omega_1 \rightarrow \omega_2) \rightsquigarrow \rho_3$, and $[\Omega]\Delta \Vdash^{kc} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_4$ and $\Delta \Vdash^{kapp} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$. Moreover, $[\Omega']\omega = \omega_2$. Further, $[\Omega']\rho_2 = \rho_3 \rho_4$.

Lemma F.66 (Principality of Typing Data Constructor Declaration). Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash_p^{dc} \mathcal{D} \rightsquigarrow \mu_1$, and $\Delta \Vdash_p^{dc} \mathcal{D} \rightsquigarrow \mu_2 \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$.

Lemma F.67 (Principality of Typing Datatype Declaration). Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \Vdash^{dt} \mathcal{T} \rightsquigarrow \Psi$, and $\Delta \Vdash^{dt} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$.

Theorem F.68 (Principality of Typing a Datatype Declaration Group). If $\Omega \Vdash^{grp} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\eta}_i^i ; \overline{\Gamma}_i^i$, then whenever $[\Omega]\Omega \Vdash^{grp} \mathbf{rec} \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\eta}'_i^i ; \overline{\Psi}_i^i$ holds, we have $[\Omega]\Omega \vdash [\Omega]\eta_i \leq \eta'_i$.

F.2 Proofs

Lemma F.1 (Well-formedness of Declarative Instantiation). If $\Sigma \Vdash^{ela} \mu_1 : \eta_1$, and $\Sigma \Vdash^{inst} \mu_1 : \eta_1 \sqsubseteq \eta_2 \rightsquigarrow \mu_2$, then $\Sigma \Vdash^{ela} \mu_2 : \eta_2$.

PROOF. By induction on the derivation.

- Case

$$\frac{\text{INST-REFL}}{\Sigma \Vdash^{inst} \mu : \omega \sqsubseteq \omega \rightsquigarrow \mu}$$

The goal follows trivially.

- Case

$$\frac{\Sigma \Vdash^{ela} \rho : \omega_1 \quad \Sigma \Vdash^{inst} \mu_1 @\rho : \eta[a \mapsto \rho] \sqsubseteq \omega_2 \rightsquigarrow \mu_2}{\Sigma \Vdash^{inst} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2}$$

$$\begin{array}{l|l} \Sigma \Vdash^{ela} \mu_1 : \forall a : \omega_1. \eta & \text{Given} \\ \Sigma \Vdash^{ela} \rho : \omega_1 & \text{Given} \\ \Sigma \Vdash^{ela} \mu_1 @\rho : \eta[a \mapsto \rho] & \text{By rule ELA-KAPP} \\ \Sigma \Vdash^{ela} \mu_2 : \eta_2 & \text{I.H.} \end{array}$$

- Case

$$\frac{\text{INST-FORALL-INFER} \quad \Sigma \Vdash^{ela} \rho : \omega_1 \quad \Sigma \Vdash^{inst} \mu_1 @\rho : \eta[a \mapsto \rho] \sqsubseteq \omega_2 \rightsquigarrow \mu_2}{\Sigma \Vdash^{inst} \mu_1 : \forall \{a : \omega_1\}. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2}$$

Similar as the previous case.

□

Lemma F.2 (Well-formedness of Declarative Kinding). We have:

- if $\Sigma \Vdash^k \sigma : \eta \rightsquigarrow \mu$, then $\Sigma \Vdash^{ela} \mu : \eta$;

- if $\Sigma \vdash^{\text{kc}} \sigma \Leftarrow \eta \rightsquigarrow \mu$, then $\Sigma \vdash^{\text{ela}} \mu : \eta$.

PROOF. By induction on the derivation.

Part 1 • Case for rules **KTT-STAR**, **KTT-NAT**, **KTT-VAR**, **KTT-TCON**, and **KTT-ARROW** holds trivially.

- Case

$$\frac{\text{KTT-APP} \quad \Sigma \vdash^{\text{k}} \tau_1 : \eta_1 \rightsquigarrow \rho_1 \quad \Sigma \vdash^{\text{inst}} \rho_1 : \eta_1 \sqsubseteq (\omega_1 \rightarrow \omega_2) \rightsquigarrow \rho_2 \quad \Sigma \vdash^{\text{kc}} \tau_2 \Leftarrow \omega_1 \rightsquigarrow \rho_3}{\Sigma \vdash^{\text{k}} \tau_1 \tau_2 : \omega_2 \rightsquigarrow \rho_2 \rho_3}$$

$$\left. \begin{array}{l} \Sigma \vdash^{\text{ela}} \rho_1 : \eta_1 \\ \Sigma \vdash^{\text{ela}} \rho_2 : \omega_1 \rightarrow \omega_2 \\ \Sigma \vdash^{\text{ela}} \rho_3 : \omega_1 \\ \Sigma \vdash^{\text{ela}} \rho_2 \rho_3 : \omega_2 \end{array} \right| \begin{array}{l} \text{I.H.} \\ \text{By Lemma F.1} \\ \text{By part 2} \\ \text{By rule ELA-APP} \end{array}$$

- The rest cases are similar, following directly from I.H. and part 2.

Part 2

$$\frac{\text{KC-SUB} \quad \Sigma \vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu_1 \quad \Sigma \vdash^{\text{inst}} \mu_1 : \eta \sqsubseteq \omega \rightsquigarrow \mu_2}{\Sigma \vdash^{\text{kc}} \sigma \Leftarrow \omega \rightsquigarrow \mu_2}$$

$$\left. \begin{array}{l} \Sigma \vdash^{\text{ela}} \mu_1 : \eta \\ \Sigma \vdash^{\text{ela}} \mu_2 : \omega \end{array} \right| \begin{array}{l} \text{By part 1} \\ \text{By Lemma F.1} \end{array}$$

□

Lemma F.3 (Well-formedness of Declarative Elaborated Kinding). *If Σ ok, and $\Sigma \vdash^{\text{ela}} \mu : \eta$, then $\Sigma \vdash^{\text{ela}} \eta : \star$.*

PROOF. By a straightforward induction on the judgment, utilizing the substitution lemma.

□

Lemma F.4 (Well-formedness of Declarative Typing Signature). *If Σ ok, and $\Sigma \vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$, then $\Sigma \vdash^{\text{ela}} \eta : \star$.*

PROOF. We have

$$\frac{\text{SIG-TT} \quad \lceil \sigma \rceil \quad \phi \in \mathcal{Q}(\sigma) \quad \phi_1^c \in \mathcal{Q}(\forall \phi^c. \eta) \quad \Sigma, \phi_1^c \vdash^{\text{k}} \forall \phi. \sigma : \star \rightsquigarrow \forall \phi^c. \eta \quad |\phi| = |\phi^c|}{\Sigma \vdash^{\text{sig}} \mathbf{data} T : \sigma \rightsquigarrow T : \forall \{\phi_1^c\}. \forall \{\phi^c\}. \eta}$$

$$\left. \begin{array}{l} \Sigma, \phi_1^c \vdash^{\text{k}} \forall \phi. \sigma : \star \rightsquigarrow \forall \phi^c. \eta \\ \Sigma, \phi_1^c \vdash^{\text{ela}} \forall \phi^c. \eta : \star \\ \Sigma, \phi_1^c \vdash^{\text{ela}} \forall \{\phi^c\}. \eta : \star \\ \phi_1^c \text{ is well-formed} \\ \Sigma \vdash^{\text{ela}} \forall \{\phi_1^c\}. \forall \{\phi^c\}. \eta : \star \end{array} \right| \begin{array}{l} \text{Given} \\ \text{By Lemma F.2} \\ \text{By rule ELA-FORALL-INFER} \\ \text{By rule ELA-FORALL-INFER} \end{array}$$

□

Lemma F.5 (Well-formedness of Declarative Typing Data Constructor Declaration). *If Σ ok, and $\Sigma \vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu$, then $\Sigma \vdash^{\text{ela}} \mu : \star$.*

PROOF. We have

$$\frac{\text{DC-TT} \quad \phi^c \in \mathcal{Q}(\mu \setminus_{\Sigma, \bar{\tau}_i^i}) \quad \Sigma, \phi^c \Vdash^k \forall \phi. \bar{\tau}_i^i \rightarrow \rho : \star \rightsquigarrow \mu}{\Sigma \Vdash_{\rho}^{\text{dc}} \forall \phi. D \bar{\tau}_i^i \rightsquigarrow \forall \{\phi^c\}. \mu}$$

$$\left. \begin{array}{l} \Sigma, \phi^c \Vdash^k \forall \phi. \bar{\tau}_i^i \rightarrow \rho : \star \rightsquigarrow \mu \\ \Sigma, \phi^c \Vdash^{\text{ela}} \mu : \star \\ \phi^c \text{ is well-formed} \\ \Sigma \Vdash^{\text{ela}} \forall \{\phi^c\}. \mu : \star \end{array} \right\} \begin{array}{l} \text{Given} \\ \text{By Lemma F.2} \\ \text{By rule ELA-FORALL-INFER} \end{array}$$

□

Lemma F.6 (Well-formedness of Declarative Typing Datatype Declaration). *If Σ ok, and $\Sigma \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, then $\Sigma \vdash \Psi$.*

PROOF. We have

$$\frac{\text{DT-TT} \quad (T : \forall \{\phi_1^c\}. \forall \phi_2^c. \bar{\omega}_i^i \rightarrow \star) \in \Sigma \quad \frac{\Sigma, \phi_1^c, \phi_2^c, \bar{a}_i : \bar{\omega}_i^i \Vdash^{\text{dc}} (T \text{ @ } \phi_1^c \text{ @ } \phi_2^c \bar{a}_i^i) \mathcal{D}_j \rightsquigarrow \mu_j^j}{\Sigma \Vdash^{\text{dt}} \mathbf{data} T \bar{a}_i^i = \bar{\mathcal{D}}_j^j \rightsquigarrow D_j : \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \bar{a}_i : \bar{\omega}_i^i. \mu_j^j}}{\Sigma \Vdash^{\text{dt}} \mathbf{data} T \bar{a}_i^i = \bar{\mathcal{D}}_j^j \rightsquigarrow D_j : \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \bar{a}_i : \bar{\omega}_i^i. \mu_j^j}$$

$$\left. \begin{array}{l} \frac{\Sigma, \phi_1^c, \phi_2^c, \bar{a}_i : \bar{\omega}_i^i \Vdash^{\text{ela}} \mu_j^j : \star}{\Sigma \Vdash^{\text{ela}} \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \bar{a}_i : \bar{\omega}_i^i. \mu_j^j : \star} \\ \Sigma \vdash D_j : \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \bar{a}_i : \bar{\omega}_i^i. \mu_j^j \end{array} \right\} \begin{array}{l} \text{By Lemma F.5} \\ \text{By rule ELA-FORALL} \\ \text{By rule ECTX-DCON} \end{array}$$

□

Lemma F.7 (Well-formedness of Declarative Generalization). *If Σ ok, and $\Sigma \vdash \Psi_1$ and $\Sigma \Vdash_{\phi^c}^{\text{gen}} \Psi_1 \rightsquigarrow \Psi_2$, then $\Sigma \vdash \Psi_2$.*

PROOF. Follows directly from rule ECTX-DCON-TT and rule ELA-FORALL-INFER.

□

F.2.1 Well-formedness of Algorithmic Type System.

Lemma F.8 (Well-formedness of Promotion). *If $\Delta_1, \hat{\alpha} : \omega, \Delta_2$ ok, and $\Delta_1, \hat{\alpha} : \omega, \Delta_2 \Vdash^{\text{ela}} \rho_1 : \omega_2$, and $\Delta_1, \hat{\alpha} : \omega, \Delta_2 \Vdash_{\hat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta$, then $\Theta = \Theta_1, \hat{\alpha} : \omega, \Theta_2$, and $\Delta_1, \hat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta$, and $\Theta_1 \Vdash^{\text{ela}} \rho_2 : [\Theta] \omega_2$, and Θ ok. By weakening, there is also $\Theta \Vdash^{\text{ela}} \rho_2 : [\Theta] \omega_2$. Similar lemma holds when in the input context, $\hat{\alpha} : \omega$ is in a local scope.*

PROOF. For most cases, the goal follows directly.

The case for rule A-PR-KAPP is similar as rule A-PR-APP.

- Case

$$\frac{\text{A-PR-APP} \quad \Delta \Vdash_{\hat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash_{\hat{\alpha}}^{\text{pr}} [\Delta_1] \omega_2 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta \Vdash_{\hat{\alpha}}^{\text{pr}} \omega_1 \omega_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}$$

$$\left. \begin{array}{l} \Delta \Vdash^{\text{ela}} \omega_1 \omega_2 : \omega'_2 \\ \Delta \Vdash^{\text{ela}} \omega_1 : \omega'_1 \rightarrow \omega'_2 \wedge \Delta \Vdash^{\text{ela}} \omega_2 : \omega'_1 \end{array} \right\} \begin{array}{l} \text{Given} \\ \text{By inversion} \end{array}$$

$\Delta \longrightarrow \Delta_1 \wedge \Delta_1 = \Delta_{11}, \widehat{\alpha} : \rho, \Delta_{12} \wedge \Delta_1 \text{ ok} \wedge \Delta_{11} \Vdash^{\text{ela}} \rho_1 : [\Delta_{11}]\omega'_1 \rightarrow [\Delta_{11}]\omega'_2$	I.H.
$\Delta_1 \Vdash^{\text{ela}} \omega_2 : [\Delta_1]\omega'_1$	By Lemma F.22
$\Delta_1 \Vdash^{\text{ela}} [\Delta_1]\omega_2 : [\Delta_1]\omega'_1$	By Lemma F.24
$\Delta_1 \longrightarrow \Theta \wedge \Theta = \Theta_1, \widehat{\alpha} : \rho, \Theta_2 \wedge \Theta \text{ ok} \wedge \Theta_1 \Vdash^{\text{ela}} \rho_2 : [\Theta]([\Delta_1]\omega'_1)$	I.H.
$\Delta_{11} \longrightarrow \Theta_1$	By Lemma F.29
$\Delta \longrightarrow \Theta$	By Lemma F.32
$\Theta_1 \Vdash^{\text{ela}} \rho_1 : [\Theta_1]([\Delta_{11}]\omega'_1) \rightarrow [\Theta_1]([\Delta_{11}]\omega'_2)$	By Lemma F.22
$\Theta_1 \Vdash^{\text{ela}} \rho_1 : [\Theta_1]\omega'_1 \rightarrow [\Theta_1]\omega'_2$	By Lemma F.30
$\Theta_1 \Vdash^{\text{ela}} \rho_2 : [\Theta]\omega'_1$	By Lemma F.30
$\Theta_1 \Vdash^{\text{ela}} \rho_1 \rho_2 : [\Theta]\omega'_2$	By rule A-ELA-APP

• Case

$$\frac{\text{A-PR-KUVR-R-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} [\Delta]\rho \rightsquigarrow \rho_1 \dashv \Theta[\widehat{\alpha}][\widehat{\beta} : \rho]}{\Delta[\widehat{\alpha}][\widehat{\beta} : \rho] \Vdash_{\widehat{\alpha}}^{\text{pr}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_1 \dashv \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]}$$

$\Delta \text{ ok}$	Given
$\Delta \Vdash^{\text{ela}} \rho : \star$	By inversion
$\Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1] = \Theta_1, \widehat{\beta}_1 : \rho_1, \widehat{\alpha} : \rho_2, \Theta_2, \widehat{\beta} : \rho = \widehat{\beta}_1, \Theta_3$	Suppose
$\Theta[\widehat{\alpha}][\widehat{\beta} : \rho] \text{ ok} \wedge \Delta \longrightarrow \Theta[\widehat{\alpha}][\widehat{\beta} : \rho] \wedge \Theta_1 \Vdash^{\text{ela}} \rho_1 : \star$	I.H.
$\Delta \longrightarrow \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho]$	By Lemma F.35, Lemma F.32
$\Theta_1, \widehat{\beta}_1 : \rho_1, \widehat{\alpha} : \rho_2, \Theta_2 \Vdash^{\text{ela}} \widehat{\beta}_1 : [\Theta_1]\rho_1$	By rule A-ELA-VAR and Lemma F.31
$\Theta_1, \widehat{\beta}_1 : \rho_1, \widehat{\alpha} : \rho_2, \Theta_2 \Vdash^{\text{ela}} \widehat{\beta}_1 : [\Theta[\widehat{\alpha}][\widehat{\beta} : \rho]]\rho_1$	By Lemma F.31
$\Theta_1, \widehat{\beta}_1 : \rho_1, \widehat{\alpha} : \rho_2, \Theta_2 \Vdash^{\text{ela}} \widehat{\beta}_1 : [\Theta[\widehat{\alpha}][\widehat{\beta} : \rho]]([\Delta]\rho)$	By Lemma F.53
$\Theta_1, \widehat{\beta}_1 : \rho_1, \widehat{\alpha} : \rho_2, \Theta_2 \Vdash^{\text{ela}} \widehat{\beta}_1 : [\Theta[\widehat{\alpha}][\widehat{\beta} : \rho]]\rho$	By Lemma F.30
$\Theta_1, \widehat{\beta}_1 : \rho_1, \widehat{\alpha} : \rho_2, \Theta_2 \Vdash^{\text{ela}} \widehat{\beta}_1 : [\Theta_1, \widehat{\alpha} : \rho_2, \Theta_2]\rho$	By Lemma F.53
$\Delta \longrightarrow \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]$	By Lemma F.33, Lemma F.32

□

Lemma F.9 (Well-formedness of Moving). *If $\Delta_1 \dashv^{\text{mv}} \Delta_2 \rightsquigarrow \Theta$, then $\text{topo}(\Delta_1, \Delta_2) = \Theta$.*

PROOF. By a straightforward induction on the moving judgment.

□

Lemma F.10 (Well-formedness of Unification). *If $\Delta \text{ ok}$, and $\Delta \Vdash^{\text{u}} \kappa_1 \approx \kappa_2 \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \text{ ok}$.*

PROOF. By induction on the derivation.

- The case for rule A-U-REFL-TT follows directly from Lemma F.26.
- Case

$$\frac{\text{A-U-KVARL-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \quad \Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1 \Vdash^{\text{u}} [\Theta_1]\omega_1 \approx \omega_2 \dashv \Theta_3}{\Delta \Vdash^{\text{u}} \widehat{\alpha} \approx \rho_1 \dashv \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2}$$

$\Delta \longrightarrow \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \wedge \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \text{ ok}$	By Lemma F.8
$\Theta_1 \longrightarrow \Theta_3$	I.H.
$\Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2$	Given
$\Theta_3 \Vdash^{\text{ela}} \rho_2 : [\Theta_3]\omega_2$	By Lemma F.22

$[\Theta_3]\omega_2 = [\Theta_3]([\Theta_1]\omega_1)$	By Lemma F.54
$[\Theta_3]\omega_2 = [\Theta_3]\omega_1$	By Lemma F.30
$\Theta_3 \Vdash^{\text{ela}} \rho_2 : [\Theta_3]\omega_1$	By substituting equations
$\Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \longrightarrow \Theta_3, \widehat{\alpha} : \omega_1, \Theta_2$	By extension rules
$\Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \longrightarrow \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2$	Lemma F.33
$\Delta \longrightarrow \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2$	Lemma F.32

- The case for rule **A-U-KVARR-TT** is similar as the previous case.
- Case

$$\frac{\begin{array}{l} \text{A-U-KVARL-LO-TT} \\ \Delta_1, \Delta_2 \Vdash^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 + \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \\ \Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2]\omega_1 \approx \omega_2 + \Theta_5, \{\Theta_6\} \end{array}}{\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 + \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4}$$

$\text{topo}(\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2) = \Theta$	by Lemma F.9
$\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \longrightarrow \Delta[\{\Theta\}]$	By definition
$\Delta[\{\Theta\}] \longrightarrow \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4$	Lemma F.8
$\Theta_1, \{\Theta_2\} \longrightarrow \Theta_5, \{\Theta_6\}$	I.H.
$\Theta_1, \text{topo}(\Theta_2) \longrightarrow \Theta_5, \Theta_6$	By inversion
$\Theta_1, \text{topo}(\Theta_2), \widehat{\alpha} : \omega_1, \Theta_3 \longrightarrow \Theta_5, \Theta_6, \widehat{\alpha} : \omega_1, \Theta_3$	By definition
$\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\} \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1, \Theta_3\}$	By rule A-CTXE-LO
$\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4$	By definition
$\Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2$	Given
$\Theta_5, \{\Theta_6\} \Vdash^{\text{ela}} \rho_2 : [\Theta_5, \Theta_6]\omega_2$	By Lemma F.22
$[\Theta_5, \Theta_6]\omega_2 = [\Theta_5, \Theta_6]([\Theta_1, \Theta_2]\omega_1)$	Lemma F.54
$[\Theta_5, \Theta_6]\omega_2 = [\Theta_5, \Theta_6]\omega_1$	Lemma F.30
$\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$	Lemma F.33
$\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$	By Lemma F.32

- The case for rule **A-U-KVARR-LO-TT** is similar as the previous case.
- The case for rule **A-U-APP** follows directly from I.H. and Lemma F.32.
- The case for rule **A-U-KAPP** follows directly from I.H. and Lemma F.32.

□

Lemma F.11 (Well-formedness of Instantiation). *If $\Delta \Vdash^{\text{inst}} \rho_1 : \eta_1 \sqsubseteq \eta_2 \rightsquigarrow \rho_2 + \Theta$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \eta_1$, then $\Delta \longrightarrow \Theta$, and Θ ok, and $\Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\eta_2$.*

PROOF. By induction on the derivation.

- Case

$$\frac{\text{A-INST-REFL} \quad \Delta \Vdash^{\mu} \omega_1 \approx \omega_2 + \Theta}{\Delta \Vdash^{\text{inst}} \mu : \omega_1 \sqsubseteq \omega_2 \rightsquigarrow \mu + \Theta}$$

$\Delta \longrightarrow \Theta$	Lemma F.10
$[\Theta]\omega_1 = [\Theta]\omega_2$	Lemma F.54
$\Delta \Vdash^{\text{ela}} \mu : \omega_1$	Given
$\Theta \Vdash^{\text{ela}} \mu : [\Theta]\omega_1$	Lemma F.22
$\Theta \Vdash^{\text{ela}} \mu : [\Theta]\omega_2$	By equations

- Case

$$\frac{\text{A-INST-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{inst}} \mu_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}] \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}{\Delta \Vdash^{\text{inst}} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}$$

$\begin{aligned} \Delta &\longrightarrow \Delta, \widehat{\alpha} : \omega_1 \\ \Delta &\Vdash^{\text{ela}} \mu_1 : \forall a : \omega_1. \eta \\ [\Delta](\forall a : \omega_1. \eta) &= \forall a : \omega_1. \eta \\ [\Delta]\omega_1 &= \omega_1 \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \widehat{\alpha} : [\Delta]\omega_1 \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \widehat{\alpha} : \omega_1 \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \mu_1 : [\Delta, \widehat{\alpha} : \omega_1](\forall a : \omega_1. \eta) \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \mu_1 : [\Delta](\forall a : \omega_1. \eta) \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \mu_1 : (\forall a : \omega_1. \eta) \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \mu_1 @ \widehat{\alpha} : \eta[a \mapsto [\Delta, \widehat{\alpha} : \omega_1]\widehat{\alpha}] \\ \Delta, \widehat{\alpha} : \omega_1 &\Vdash^{\text{ela}} \mu_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}] \\ \Delta, \widehat{\alpha} : \omega_1 &\longrightarrow \Theta \wedge \Theta \Vdash^{\text{ela}} \mu_2 : [\Theta]\omega_2 \\ \Delta &\longrightarrow \Theta \end{aligned}$	<div style="border-left: 1px solid black; padding-left: 10px;"> <p>rule A-CTXE-ADD-TT</p> <p>Given</p> <p>By Lemma F.16</p> <p>By inversion</p> <p>By rule A-ELA-KUVAR</p> <p>By equation</p> <p>By Lemma F.22</p> <p>$\widehat{\alpha}$ fresh</p> <p>by equation</p> <p>By rule A-ELA-KAPP</p> <p>By definition</p> <p>I.H.</p> <p>Lemma F.32</p> </div>
--	--

- The case for rule **A-INST-FORALL-INFER** is similar to the previous case. □

Lemma F.12 (Well-formedness of Quantification Check). *If $\Delta_1, a : \omega, \Delta_2$ ok, and $\Delta_2 \hookrightarrow a$, then Δ_1, Δ_2 ok.*

PROOF. All items in Δ_2 are well-formed by strengthening on elaborated kinding. □

Lemma F.13 (Well-formedness of Unsolved). *If Δ_1, Δ_2 ok, and Δ_2 soft, then $\Delta_1, \text{unsolved}(\Delta_2)$ ok.*

PROOF. All unification variables in $\text{unsolved}(\Delta_2)$ are well-formed, which can be derived similarly as Lemma **F.24**, and strengthening on elaborated kinding. □

Lemma F.14 (Well-formedness of topo). *If Δ_1, Δ_2 ok, then $\Delta_1, \text{topo}(\Delta_2)$ ok.*

PROOF. As $\Delta_1, \text{topo}(\Delta_2)$ ok preserves a well-formed ordering, by strengthening and weakening we can prove $\Delta_1, \text{topo}(\Delta_2)$ ok. □

Lemma F.15 (Well-formedness of Kinding). *Given Δ ok,*

- if $\Delta \Vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok and $\Theta \Vdash^{\text{ela}} \mu : [\Theta]\eta$;
- if $\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \eta \rightsquigarrow \mu \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and Θ ok and $\Theta \Vdash^{\text{ela}} \mu : [\Theta]\eta$.
- if $\Delta \Vdash^{\text{kapp}} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 \dashv \Theta$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \eta$, then $\Delta \longrightarrow \Theta$, and Θ ok, and $\Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\omega$.

PROOF. By induction on the derivation.

Part 1 • The case for rules **A-KTT-STAR**, **A-KTT-NAT**, **A-KTT-VAR**, **A-KTT-TCON**, and **A-KTT-ARROW** is trivial.

- Case

$$\frac{\text{A-KTT-FORALL} \quad \Delta \Vdash^{\text{kc}} \kappa \Leftarrow \star \rightsquigarrow \omega \dashv \Delta_1 \quad \Delta_1, a : \omega \Vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow \mu \dashv \Delta_2, a : \omega, \Delta_3 \quad \Delta_3 \hookrightarrow a}{\Delta \Vdash^{\text{k}} \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega. [\Delta_3]\mu \dashv \Delta_2, \text{unsolved}(\Delta_3)}$$

$\Delta \longrightarrow \Delta_1 \wedge \Delta \Vdash^{\text{ela}} \omega : \star \wedge \Delta_1 \text{ ok}$	Part 2
$\Delta_1 \Vdash^{\text{ela}} \omega : \star$	By Lemma F.22
$\Delta_1, a : \omega \text{ ok}$	By rule A-TCTX-TVAR-TT
$\Delta_1, a : \omega \longrightarrow \Delta_2, a : \omega, \Delta_3 \wedge \Delta_2, a : \omega, \Delta_3 \text{ ok}$	Part 2
$\wedge \Delta_2, a : \omega, \Delta_3 \Vdash^{\text{ela}} \mu : \star$	Above
$\Delta_1 \longrightarrow \Delta_2 \wedge \Delta_3 \text{ soft}$	By Lemma F.29
$\Delta_3 \hookrightarrow a$	Given
$\Delta_2, \Delta_3 \text{ ok}$	By Lemma F.12
$\Delta_2, \text{unsolved}(\Delta_3) \text{ ok}$	By Lemma F.13
$\Delta_2 \longrightarrow \Delta_2, \text{unsolved}(\Delta_3)$	By rules A-CTXE-ADD-TT and A-CTXE-ADDSOLVED-TT
$\Delta \longrightarrow \Delta_2, \text{unsolved}(\Delta_3)$	Lemma F.32
$\Delta_2, \text{unsolved}(\Delta_3) \Vdash^{\text{ela}} \omega : \star.$	Lemma F.22
$\Delta_2, a : \omega, \Delta_3 \Vdash^{\text{ela}} \mu : \star$	Known
$\Delta_2, a : \omega, \text{unsolved}(\Delta_3) \Vdash^{\text{ela}} [\Delta_3]\mu : \star.$	By Lemma F.25

Because $\text{unsolved}(\Delta_3)$ does not depend on a , we can reorder the context to get that $\Delta_2, \text{unsolved}(\Delta_3), a \Vdash^{\text{ela}} [\Delta_3]\mu : \star$. So by rule A-ELA-FORALL we get $\Delta_2, \text{unsolved}(\Delta_3) \Vdash^{\text{ela}} \forall a : \omega. [\Delta_3]\mu : \star$.

- The case for rule A-KTT-FORALLI is similar as the previous case.
- Case

$$\frac{\text{A-KTT-APP} \quad \Delta \Vdash^{\text{k}} \tau_1 : \eta_1 \rightsquigarrow \rho_1 \div \Delta_1 \quad \Delta_1 \Vdash^{\text{kapp}} (\rho_1 : [\Delta_1]\eta_1) \bullet \tau_2 : \omega \rightsquigarrow \rho \div \Theta}{\Delta \Vdash^{\text{k}} \tau_1 \tau_2 : \omega \rightsquigarrow \rho \div \Theta}$$

$\Delta \longrightarrow \Delta_1 \wedge \Delta_1 \Vdash^{\text{ela}} \rho_1 : [\Delta_1]\eta_1$	I.H.
$\Delta_1 \longrightarrow \Theta \wedge \Theta \Vdash^{\text{ela}} \rho : [\Theta]\omega$	Part 3
$\Delta \longrightarrow \Theta$	By Lemma F.32

- Case

$$\frac{\text{A-KTT-KAPP} \quad \Delta \Vdash^{\text{k}} \tau_1 : \eta \rightsquigarrow \rho_1 \div \Delta_1 \quad [\Delta_1]\eta = \forall a : \omega. \eta_2 \quad \Delta_1 \Vdash^{\text{kcc}} \tau_2 \Leftarrow \omega \rightsquigarrow \rho_2 \div \Delta_2}{\Delta \Vdash^{\text{k}} \tau_1 @ \tau_2 : \eta_2[a \mapsto \rho_2] \rightsquigarrow \rho_1 @ \rho_2 \div \Delta_2}$$

$\Delta \longrightarrow \Delta_1 \wedge \Delta_1 \Vdash^{\text{ela}} \rho_1 : [\Delta_1]\eta$	I.H.
$\Delta_1 \longrightarrow \Delta_2 \wedge \Delta_2 \Vdash^{\text{ela}} \rho_2 : [\Delta_2]\omega$	Part 2
$\Delta \longrightarrow \Delta_2$	Lemma F.32
$\Delta_1 \Vdash^{\text{ela}} \rho_1 : \forall a : \omega. \eta_2$	by equations
$\Delta_2 \Vdash^{\text{ela}} \rho_1 : \forall a : [\Delta_2]\omega. [\Delta_2]\eta_2$	Lemma F.22
$\Delta_2 \Vdash^{\text{ela}} \rho_1 @ \rho_2 : ([\Delta_2]\eta_2)[a \mapsto [\Delta_2]\rho_2]$	By rule A-ELA-KAPP
$\Delta_2 \Vdash^{\text{ela}} \rho_1 @ \rho_2 : [\Delta_2](\eta_2[a \mapsto \rho_2])$	By substitution

- Case rule A-KTT-KAPP-INFER is similar as the previous case.

Part 2 We have

$$\frac{\text{A-KC-SUB} \quad \Delta \Vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu_1 \div \Delta_1 \quad \Delta_1 \Vdash^{\text{inst}} \mu_1 : [\Delta_1]\eta \sqsubseteq [\Delta_1]\omega \rightsquigarrow \mu_2 \div \Delta_2}{\Delta \Vdash^{\text{kcc}} \sigma \Leftarrow \omega \rightsquigarrow \mu_2 \div \Delta_2}$$

$\Delta \longrightarrow \Delta_1 \wedge \Delta_1 \Vdash^{\text{ela}} \mu_1 : [\Delta_1]\eta$	Part 1
$\Delta_1 \longrightarrow \Delta_2 \wedge \Delta_2 \Vdash^{\text{ela}} \mu_2 : [\Delta_2]([\Delta_1]\omega_2)$	By Lemma F.11
$\Delta \longrightarrow \Delta_2$	Lemma F.32

$$\begin{array}{l} [\Delta_2]([\Delta_1]\omega) = [\Delta_2]\omega \\ \Delta_2 \Vdash^{\text{ela}} \mu_2 : [\Delta_2]\omega \end{array} \quad \left| \begin{array}{l} \text{Lemma F.30} \\ \text{by equations} \end{array} \right.$$

Part 3 By induction on the judgment.

- Case

$$\frac{\text{A-KAPP-TT-ARROW} \quad \Delta \Vdash^{\text{kc}} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \omega_1 \rightarrow \omega_2) \bullet \tau : \omega_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}$$

$$\begin{array}{l} \Delta \longrightarrow \Theta \wedge \Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\omega_1 \\ \Delta \Vdash^{\text{ela}} \rho_1 : \omega_1 \rightarrow \omega_2 \\ \Theta \Vdash^{\text{ela}} \rho_1 : [\Theta]\omega_1 \rightarrow [\Theta]\omega_2 \\ \Theta \Vdash^{\text{ela}} \rho_1 \rho_2 : [\Theta]\omega_2 \end{array} \quad \left| \begin{array}{l} \text{By Part 2} \\ \text{Given} \\ \text{By Lemma F.22} \\ \text{By rule A-ELA-APP} \end{array} \right.$$

- Case

$$\frac{\text{A-KAPP-TT-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{kapp}} (\rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}]) \bullet \tau : \omega \rightsquigarrow \rho \dashv \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \forall a : \omega_1. \eta) \bullet \tau : \omega \rightsquigarrow \rho \dashv \Theta}$$

$$\begin{array}{l} \Delta \Vdash^{\text{ela}} \rho_1 : \forall a : \omega_1. \eta \\ \Delta \Vdash^{\text{ela}} \forall a : \omega_1. \eta : \star \\ \Delta \Vdash^{\text{ela}} \omega_1 : \star \\ \Delta \longrightarrow \Delta, \widehat{\alpha} : \omega_1 \\ \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{ela}} \rho_1 : \forall a : \omega_1. \eta \\ \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{ela}} \widehat{\alpha} : \omega_1 \\ \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{ela}} \rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}] \\ \Delta, \widehat{\alpha} : \omega_1 \longrightarrow \Theta \wedge \Theta \Vdash^{\text{ela}} \rho : [\Theta]\omega \\ \Delta \longrightarrow \Theta \end{array} \quad \left| \begin{array}{l} \text{Given} \\ \text{By Lemma F.16} \\ \text{By inversion} \\ \text{By rule A-CTXE-ADD-TT} \\ \text{By Lemma F.22 and } \widehat{\alpha} \text{ fresh} \\ \text{By rule A-ELA-KUVAR, Lemma F.16, and } \widehat{\alpha} \text{ fresh} \\ \text{By rule A-ELA-KAPP} \\ \text{I.H.} \\ \text{By Lemma F.32} \end{array} \right.$$

- The case for rule **A-KAPP-TT-FORALL-INFER** is similar to the previous case.
- Case

$$\frac{\text{A-KAPP-TT-KUVAR} \quad \Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \Vdash^{\text{kc}} \tau \Leftarrow \widehat{\alpha}_1 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash^{\text{kapp}} (\rho_1 : \widehat{\alpha}) \bullet \tau : \widehat{\alpha}_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}$$

$$\begin{array}{l} \Delta_1, \widehat{\alpha} : \kappa, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \kappa = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \\ \Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \kappa = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \longrightarrow \Theta \wedge \Theta \Vdash^{\text{ela}} \rho_2 : [\Theta]\widehat{\alpha}_1 \\ \Delta_1, \widehat{\alpha} : \kappa, \Delta_2 \longrightarrow \Theta \\ \Delta_1, \widehat{\alpha} : \kappa, \Delta_2 \Vdash^{\text{ela}} \rho_1 : \widehat{\alpha} \\ \Theta \Vdash^{\text{ela}} \rho_1 : [\Theta]\widehat{\alpha} \\ \Theta \Vdash^{\text{ela}} \rho_1 : [\Theta]\widehat{\alpha}_1 \rightarrow [\Theta]\widehat{\alpha}_2 \\ \Theta \Vdash^{\text{ela}} \rho_1 \rho_2 : [\Theta]\widehat{\alpha}_2 \end{array} \quad \left| \begin{array}{l} \text{By Lemmas F.32, F.33 and F.35} \\ \text{By Part 2} \\ \text{By Lemma F.32} \\ \text{Given} \\ \text{By Lemma F.22} \\ \text{By Lemma F.30} \\ \text{By rule A-ELA-APP} \end{array} \right.$$

□

Lemma F.16 (Well-formedness of Elaborated Kinding). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu : \eta$, then $\Delta \Vdash^{\text{ela}} \eta : \star$, and $[\Delta]\eta = \eta$.*

PROOF. By induction on the derivation.

- The case for rules **A-ELA-STAR**, **A-ELA-NAT**, **A-ELA-ARROW**, **A-ELA-FORALL**, and **A-ELA-FORALL-INFER** is straightforward.
- The case for rules **A-ELA-KUVAR**, **A-ELA-VAR**, and **A-ELA-TCON** is similar. Consider

$$\frac{\text{A-ELA-KUVAR} \quad (\widehat{\alpha} : \omega) \in \Delta}{\Delta \Vdash^{\text{ela}} \widehat{\alpha} : [\Delta]\omega}$$

Given Δ ok, by inversion and weakening, we have $\Delta \Vdash^{\text{ela}} \omega : \star$. By Lemma F.24, we have $\Delta \Vdash^{\text{ela}} [\Delta]\omega : \star$. And $[\Delta]([\Delta]\omega) = [\Delta]\omega$.

- Case

$$\frac{\text{A-ELA-APP} \quad \Delta \Vdash^{\text{ela}} \rho_1 : \omega_1 \rightarrow \omega_2 \quad \Delta \Vdash^{\text{ela}} \rho_2 : \omega_1}{\Delta \Vdash^{\text{ela}} \rho_1 \rho_2 : \omega_2}$$

$$\left. \begin{array}{l} \Delta \Vdash^{\text{ela}} \omega_1 \rightarrow \omega_2 : \star \wedge [\Delta](\omega_1 \rightarrow \omega_2) = \omega_1 \rightarrow \omega_2 \\ \Delta \Vdash^{\text{ela}} \omega_2 : \star \wedge [\Delta]\omega_2 = \omega_2 \end{array} \right| \begin{array}{l} \text{I.H.} \\ \text{inversion} \end{array}$$

- The case for rules **A-ELA-KAPP** and **A-ELA-KAPP-INFER** is similar. Consider

$$\frac{\text{A-ELA-KAPP} \quad \Delta \Vdash^{\text{ela}} \rho_1 : \forall a : \omega. \eta \quad \Delta \Vdash^{\text{ela}} \rho_2 : \omega}{\Delta \Vdash^{\text{ela}} \rho_1 @ \rho_2 : \eta[a \mapsto [\Delta]\rho_2]}$$

$$\left. \begin{array}{l} \Delta \Vdash^{\text{ela}} \forall a : \omega. \eta : \star \\ \Delta, a : \omega \Vdash^{\text{ela}} \eta : \star \\ \Delta \Vdash^{\text{ela}} \rho_2 : \omega \\ \Delta \Vdash^{\text{ela}} [\Delta]\rho_2 : \omega \\ \Sigma \Vdash^{\text{ela}} \eta[a \mapsto [\Delta]\rho_2] : \star \\ [\Delta](\forall a : \omega. \eta) = \forall a : \omega. \eta \\ [\Delta]\eta = \eta \\ [\Delta](\eta[a \mapsto [\Delta]\rho_2]) \\ = ([\Delta]\eta)[a \mapsto ([\Delta]([\Delta]\rho_2))] \\ = \eta[a \mapsto [\Delta]\rho_2] \end{array} \right| \begin{array}{l} \text{I.H.} \\ \text{By inversion} \\ \text{Given} \\ \text{Lemma F.24} \\ \text{By substitution} \\ \text{by I.H.} \\ \text{Follows directly} \end{array}$$

□

Lemma F.17 (Well-formedness of Typing Signature). *If Ω ok, and $\Omega \Vdash^{\text{sig}} S \rightsquigarrow T : \mu$, then $\Omega \Vdash^{\text{ela}} \mu : \star$.*

PROOF. We have

$$\frac{\text{A-SIG-TT} \quad \begin{array}{l} \uparrow \sigma \mid \quad \overline{a_i^i} = \text{fkv}(\sigma) \quad \Omega, \{\overline{\widehat{\alpha}_i : \star, a_i : \widehat{\alpha}_i^i}\} \Vdash^k \sigma : \star \rightsquigarrow \eta \dashv \Delta \\ \phi_1^c = \text{scoped_sort}(\overline{a_i : [\Delta]\widehat{\alpha}_i^i}) \quad \widehat{\phi}_2^c = \text{unsolved}(\Delta) \quad \Delta \hookrightarrow \overline{a_i^i} \end{array}}{\Omega \Vdash^{\text{sig}} \text{data } T : \sigma \rightsquigarrow T : \forall \{\phi_2^c\}. ((\forall \{\phi_1^c\}. [\Delta]\eta)[\widehat{\phi}_2^c \mapsto \phi_2^c])}$$

$$\left. \begin{array}{l} \Delta \Vdash^{\text{ela}} \eta : \star \wedge \Omega \longrightarrow \Delta \\ \Delta \Vdash^{\text{ela}} [\Delta]\eta : \star \\ \Delta = \Omega, \{\Delta_1\}, \Delta_2 \wedge \Delta_2 \text{ soft} \\ \Delta_1 \text{ contains only unification variables except for } \overline{a_i^i} \\ \Delta \hookrightarrow \overline{a_i^i} \end{array} \right| \begin{array}{l} \text{By Lemma F.15} \\ \text{By Lemma F.24} \\ \text{By Lemma F.29 and properties of kinding} \\ \text{Above} \\ \text{Given} \end{array}$$

$\Omega, \widehat{\phi}_2^c \text{ ok}$	By Lemma F.12 and Lemma F.13 ϕ_1^c has no solved unification variables By strengthening and reorder of context By rule A-ELA-FORALL-INFER By substitution By rule A-ELA-FORALL-INFER
$\Omega, \widehat{\phi}_2^c, \phi_1^c \text{ ok}$	
$\Omega, \widehat{\phi}_2^c, \phi_1^c \Vdash^{\text{ela}} [\Delta]\eta : \star$	
$\Omega, \widehat{\phi}_2^c \Vdash^{\text{ela}} \forall \{\phi_1^c\}. [\Delta]\eta : \star$	
$\Omega, \phi_2^c \Vdash^{\text{ela}} (\forall \{\phi_1^c\}. [\Delta]\eta)[\widehat{\phi}_2^c \mapsto \phi_2^c] : \star$	
$\Omega \Vdash^{\text{ela}} \forall \{\phi_2^c\}. (\forall \{\phi_1^c\}. [\Delta]\eta)[\widehat{\phi}_2^c \mapsto \phi_2^c] : \star$	

□

Lemma F.18 (Well-formedness of Typing Data Constructor Declaration). *If $\Delta \text{ ok}$, and $\Delta \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \Vdash^{\text{ela}} \mu : \star$.*

PROOF. We have

$$\frac{\text{A-DC-TT} \quad \Delta, \blacktriangleright_D \Vdash^k \forall \phi. (\overline{\tau}_i^i \rightarrow \rho) : \star \rightsquigarrow \mu \dashv \Theta_1, \blacktriangleright_D, \Theta_2 \quad \widehat{\phi}^c = \text{unsolved}(\Theta_2)}{\Delta \Vdash_{\rho}^{\text{dc}} \forall \phi. D \overline{\tau}_i^i \rightsquigarrow \forall \{\phi^c\}. ([\Theta_2]\mu)[\widehat{\phi}^c \mapsto \phi^c] \dashv \Theta_1}$$

$\Delta, \blacktriangleright_D \text{ ok}$	By rule A-TCTX-MARKER By Lemma F.15 By Lemma F.29 By Lemma F.13 By Lemma F.25 By substitution By rule A-ELA-FORALL-INFER By strengthening
$\Delta, \blacktriangleright_D \longrightarrow \Theta_1, \blacktriangleright_D, \Theta_2 \wedge \Theta_1, \blacktriangleright_D, \Theta_2 \Vdash^{\text{ela}} \mu : \star$	
$\Delta \longrightarrow \Theta_1 \wedge \Theta_2 \text{ soft}$	
$\Theta_1, \blacktriangleright_D, \widehat{\phi}^c \text{ ok}$	
$\Theta_1, \blacktriangleright_D, \widehat{\phi}^c \Vdash^{\text{ela}} [\Theta_2]\mu : \star$	
$\Theta_1, \blacktriangleright_D, \phi^c \Vdash^{\text{ela}} ([\Theta_2]\mu)[\widehat{\phi}^c \mapsto \phi^c] : \star$	
$\Theta_1, \blacktriangleright_D \Vdash^{\text{ela}} \forall \{\phi^c\}. ([\Theta_2]\mu)[\widehat{\phi}^c \mapsto \phi^c] : \star$	
$\Theta_1 \Vdash^{\text{ela}} \forall \{\phi^c\}. ([\Theta_2]\mu)[\widehat{\phi}^c \mapsto \phi^c] : \star$	

□

Lemma F.19 (Well-formedness of Typing Datatype Declaration). *If $\Delta \text{ ok}$, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, then $\Delta \longrightarrow \Theta$, and $\Theta \Vdash^{\text{ectx}} \Gamma$.*

PROOF. We have

$$\frac{\text{A-DT-TT} \quad (T : \forall \{\phi_1^c\}. \forall \phi_2^c. \omega) \in \Delta \quad \Delta, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star \Vdash^{\mu} [\Delta]\omega \approx (\overline{\alpha}_i^i \rightarrow \star) \dashv \Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \omega_i^i}{\frac{\Theta_j, \phi_1^c, \phi_2^c, \overline{a}_i : \omega_i^i \Vdash^{\text{dc}}_{(T \text{ @ } \phi_1^c \text{ @ } \phi_2^c \text{ @ } \overline{a}_i^i)} \mathcal{D}_j \rightsquigarrow \mu_j \dashv \Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{a}_i : \omega_i^i}{\Delta \Vdash^{\text{dt}} \text{data } T \overline{a}_i^i = \overline{\mathcal{D}_j}^{j \in 1..n} \rightsquigarrow \overline{D}_j : \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \overline{a}_i : \omega_i^i. \mu_j^j \dashv \Theta_{n+1}}}$$

$\Delta \text{ ok} \wedge (T : \forall \{\phi_1^c\}. \forall \phi_2^c. \omega) \in \Delta$	Given By inversion and weakening By rule A-TCTX-KUVAR-TT Lemma F.10 By Lemma F.18 By Lemma F.18 By Lemma F.29 and Lemma F.32
$\Delta, \phi_1^c, \phi_2^c \text{ ok}$	
$\Delta, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star \text{ ok}$	
$\Delta, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star \longrightarrow \Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \omega_i^i$	
$\Theta_j, \phi_1^c, \phi_2^c, \overline{a}_i : \omega_i^i \longrightarrow \Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{a}_i : \omega_i^i$	
$\Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{a}_i : \omega_i^i \Vdash^{\text{ela}} \mu_j : \star$	
$\Delta \longrightarrow \Theta_{n+1}$	

$$\frac{\frac{\Theta_{n+1}, \phi_1^c, \phi_2^c, \overline{a_i : \omega_i^i} \Vdash^{\text{ela}} \mu_j : \star}{\Theta_{n+1} \Vdash^{\text{ela}} \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \overline{a_i : \omega_i^i}. \mu_j : \star}}{\Theta_{n+1} \Vdash^{\text{ectx}} \overline{D_j : \forall \phi^c. \mu_j^j}}}{\Theta_{n+1}, \phi_1^c, \phi_2^c, \overline{a_i : \omega_i^i} \Vdash^{\text{ela}} \mu_j : \star} \quad \left| \begin{array}{l} \text{By Lemma F.22} \\ \text{By rules A-ELA-FORALL and A-ELA-FORALL-INFER} \\ \text{By rule A-ECTX-DCON-TT} \end{array} \right.$$

□

Lemma F.20 (Well-formedness of Generalization). *If Δ ok, and $\Delta \Vdash^{\text{ectx}} \Gamma_1$, and $\Delta \Vdash_{\phi^c}^{\text{gen}} \Gamma_1 \rightsquigarrow \Gamma_2$, then $\Delta \Vdash^{\text{ectx}} \Gamma_2$.*

PROOF. Follows directly from rule A-ECTX-DCON-TT, rule A-ELA-FORALL-INFER and substitution. □

F.2.2 Properties of Context Extension. Proofs for many lemmas are essentially the same as its corresponding lemmas in Haskell98. Therefore in this section, we only give proof for those of lemmas with slightly different reasoning or extra cases that are worth attention.

Lemma F.22 (Extension Weakening). *Given $\Delta \longrightarrow \Theta$, if $\Delta \Vdash^{\text{ela}} \mu : \eta$, then $\Theta \Vdash^{\text{ela}} \mu : [\Theta]\eta$.*

PROOF. By a straightforward induction on the elaborated kinding, making use of Lemma F.30. □

Lemma F.25 (Soft Substitution Kinding). *If Δ_1, Δ_2 ok, and Δ_2 soft, and $\Delta_1, \Delta_2 \Vdash^{\text{ela}} \mu : \eta$, then $\Delta_1, \text{unsolved}(\Delta_2) \Vdash^{\text{ela}} [\Delta_2]\mu : \eta$.*

PROOF. Similar as the proof for Lemma D.12, making use of weakening. □

Lemma F.27 (Well-formedness of Context Extension). *If Δ ok, and $\Delta \longrightarrow \Theta$, then Θ ok.*

PROOF. Similar as the proof for Lemma D.15.

For the case

$$\frac{\text{A-CTXE-LO} \quad \Delta \longrightarrow \Theta \quad \Delta, \text{topo}(\Delta_1) \longrightarrow \Theta, \Theta_1}{\Delta, \{\Delta_1\} \longrightarrow \Theta, \{\Theta_1\}}$$

$$\left. \begin{array}{l} \Delta, \{\Delta_1\} \text{ ok} \\ \Delta, \Delta_1 \text{ ok} \\ \Delta, \text{topo}(\Delta_1) \text{ ok} \\ \Theta, \Theta_1 \text{ ok} \end{array} \right| \begin{array}{l} \text{Given} \\ \text{By inversion} \\ \text{By Lemma F.14} \\ \text{I.H.} \end{array}$$

□

Lemma F.32 (Transitivity of Context Extension). *If Δ' ok, and $\Delta' \longrightarrow \Delta$, and $\Delta \longrightarrow \Theta$, then $\Delta' \longrightarrow \Theta$.*

PROOF. By induction on $\Delta \longrightarrow \Theta$. The proof is similar as the proof for Lemma D.20.

For the case

$$\frac{\text{A-CTXE-LO} \quad \Delta \longrightarrow \Theta \quad \Delta, \text{topo}(\Delta_1) \longrightarrow \Theta, \Theta_1}{\Delta, \{\Delta_1\} \longrightarrow \Theta, \{\Theta_1\}}$$

$\Delta' \longrightarrow \Delta, \{\Delta_1\}$	Given
$\Delta' = \Delta_2, \{\Delta_3\} \wedge \Delta_2 \longrightarrow \Delta \wedge \Delta_2, \text{topo}(\Delta_3) \longrightarrow \Delta, \Delta_1$	By inversion
$\Delta_2, \text{topo}(\Delta_3) \longrightarrow \Delta, \text{topo}(\Delta_1)$	By reordering Δ_3 according to $\text{topo}(\Delta_1)$
$\Delta_2, \text{topo}(\Delta_3) \longrightarrow \Theta, \Theta_1$	I.H.
$\Delta_2, \{\Delta_3\} \longrightarrow \Theta, \{\Theta_1\}$	By rule A-CTXE-LO

□

Lemma F.33 (Solution Admissibility for Extension).

- If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2$ ok and $\Delta_1 \Vdash^{\text{ela}} \rho : [\Delta_1]\omega$, then $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Delta_1, \widehat{\alpha} : \omega = \rho, \Delta_2$.
- If $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2$ ok and $\Delta_1, \Delta_3 \Vdash^{\text{ela}} \rho : [\Delta_1, \Delta_3]\omega$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho, \Delta_4\}, \Delta_2$.

PROOF. **Part 1** By induction on Δ_2 . The proof is similar as the proof for Lemma D.21.

For the case $\Delta_2 = \Delta'_2, \{\Delta_3\}$. By I.H., we have $\Delta_1, \widehat{\alpha} : \omega \longrightarrow \Delta_1, \widehat{\alpha} : \omega = \rho$. Then by rule **A-CTXE-LO** we have $\Delta_1, \widehat{\alpha} : \omega, \{\Delta'_2\} \longrightarrow \Delta_1, \widehat{\alpha} : \omega = \rho, \{\Delta'_2\}$.

Part 2 By induction on Δ_2 . Most cases are similar as Part 1. When Δ_2 is empty, we only need to prove $\Delta_1, \Delta_3, \widehat{\alpha} : \omega, \Delta_4 \longrightarrow \Delta_1, \Delta_3, \widehat{\alpha} : \omega = \rho, \Delta_4$. By referring Part 1 we are done.

□

Lemma F.36 (Parallel Admissibility).

- If $\Delta_1 \longrightarrow \Theta_1$, and Δ_1, Δ_2 ok, and $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$, and Δ_2 is fresh w.r.t. Θ_1 , then:
 - if $\Delta_1 \Vdash^{\text{ela}} \omega : \star$, then $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega, \Theta_2$;
 - if $\Theta_1 \Vdash^{\text{ela}} \rho : [\Theta_1]\omega$, then $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho, \Theta_2$;
 - if $[\Theta_1]\rho_1 = [\Theta_1]\rho_2$, then $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_2$.
- If $\Delta_1, \{\Delta_3\} \longrightarrow \Theta_1, \{\Theta_3\}$, and $\Delta_1, \{\Delta_3, \Delta_4\}, \Delta_2$ ok, and $\Delta_1, \{\Delta_3, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \Theta_4\}, \Theta_2$, and Δ_2, Δ_4 is fresh w.r.t. Θ_1, Θ_3 , then:
 - if $\Delta_1, \{\Delta_3\} \Vdash^{\text{ela}} \omega : \star$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega, \Theta_4\}, \Theta_2$;
 - if $\Theta_1, \{\Theta_3\} \Vdash^{\text{ela}} \rho : [\Theta_1, \Theta_3]\omega$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho, \Theta_4\}, \Theta_2$;
 - if $[\Theta_1, \Theta_3]\rho_1 = [\Theta_1, \Theta_3]\rho_2$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$.

PROOF. **Part 1** By induction on the size of Θ_2 . Most cases are similar as in Lemma D.24.

For the case where $\Theta_2 = \Theta_3, \{\Theta_4\}$, the derivation of $\Delta_1, \Delta_2 \longrightarrow \Theta_1, \Theta_2$ must conclude with rule **A-CTXE-LO**. It must be $\Delta_2 = \Delta_{21}, \{\Delta_{22}\}$.

$\Delta_1, \Delta_{21}, \{\Delta_{22}\} \longrightarrow \Theta_1, \Theta_3, \{\Theta_4\}$	Given
$\Delta_1, \Delta_{21}, \text{topo}(\Delta_{22}) \longrightarrow \Theta_1, \Theta_3, \Theta_4$	By inversion
$\Delta_1, \widehat{\alpha} : \omega, \Delta_{21}, \text{topo}(\Delta_2) \longrightarrow \Theta_1, \widehat{\alpha} : \omega, \Theta_3, \Theta_4$	I.H.
$\Delta_1, \widehat{\alpha} : \omega, \Delta_{21}, \{\Delta_2\} \longrightarrow \Theta_1, \widehat{\alpha} : \omega, \Theta_3, \{\Theta_4\}$	By rule A-CTXE-LO

Part 2 By induction on Θ_2 . Most cases are similar to Part 1. For the first case, when Θ_2 is empty, we know Δ_2 is empty. We have $\Delta_1, \text{topo}(\Delta_3), \text{topo}(\Delta_4) \longrightarrow \Theta_1, \Theta_3, \Theta_4$. By Part 1 we know $\Delta_1, \text{topo}(\Delta_3), \widehat{\alpha} : \omega, \text{topo}(\Delta_4) \longrightarrow \Theta_1, \Theta_3, \widehat{\alpha} : \omega, \Theta_4$. By rule **A-CTXE-LO** we have $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\} \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega, \Theta_4\}$.

□

Lemma F.37 (Parallel Extension Solution).

- If $\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_2$, and $[\Theta_1]\rho_1 = [\Theta_1]\rho_2$, then $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_2 \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \kappa_2, \Theta_2$.

- If $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$, and $[\Theta_1, \Theta_3]\rho_1 = [\Theta_1, \Theta_3]\rho_2$, then $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4\}, \Delta_2 \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}, \Theta_2$.

PROOF. **Part 1** By induction on Θ_2 . The proof is similar to Lemma D.25. For the case when $\Theta_2 = \Theta_3, \{\Theta_4\}$, we have $\Delta_2 = \Delta_3, \{\Delta_4\}$. And $\Delta_1, \widehat{\alpha} : \omega, \Delta_3, \{\Delta_4\} \longrightarrow \Theta_1, \widehat{\alpha} : \omega, \Theta_3, \{\Theta_4\}$. By inversion, we have $\Delta_1, \widehat{\alpha} : \omega, \Delta_3, \text{topo}(\Delta_4) \longrightarrow \Theta_1, \widehat{\alpha} : \omega, \Theta_3, \Theta_4$. By I.H., we have $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_3, \text{topo}(\Delta_4) \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_3, \Theta_4$. By rule **A-CTXE-LO** we have $\Delta_1, \widehat{\alpha} : \omega = \rho_1, \Delta_3, \{\Delta_4\} \longrightarrow \Theta_1, \widehat{\alpha} : \omega = \rho_2, \Theta_3, \{\Theta_4\}$.

Part 2 By induction on Θ_2 . Most cases are similar to Part 1. We discuss when Θ_2 is empty. Then Δ_2 must to empty. From givens we know that $\Delta_1, \Delta_5, \widehat{\alpha} : \omega, \Delta_6 \longrightarrow \Theta_1, \Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4$ where $\Delta_5, \widehat{\alpha} : \omega, \Delta_6 = \text{topo}(\Delta_3, \widehat{\alpha} : \omega, \Delta_4)$. By Part 1 we have $\Delta_1, \Delta_5, \widehat{\alpha} : \omega = \rho_1, \Delta_6 \longrightarrow \Theta_1, \Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4$. Since $\Delta_5, \widehat{\alpha} : \omega = \rho_1, \Delta_6 = \text{topo}(\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4)$, by rule **A-CTXE-LO** we have $\Delta_1, \{\Delta_3, \widehat{\alpha} : \omega = \rho_1, \Delta_4\} \longrightarrow \Theta_1, \{\Theta_3, \widehat{\alpha} : \omega = \rho_2, \Theta_4\}$. □

F.2.3 Properties of Complete Context.

Lemma F.43 (Stability of Complete Contexts). *If $\Delta \longrightarrow \Omega$, then $[\Omega]\Delta = [\Omega]\Omega$.*

PROOF. By induction on $\Delta \longrightarrow \Omega$. Most cases are the same as Lemma D.31. For the case

$$\frac{\text{A-CTXE-LO} \quad \Delta \longrightarrow \Theta \quad \Delta, \text{topo}(\Delta_1) \longrightarrow \Theta, \Theta_1}{\Delta, \{\Delta_1\} \longrightarrow \Theta, \{\Theta_1\}}$$

$\Delta, \{\Delta_1\} \longrightarrow \Omega, \{\Omega_1\}$	Given
$\Delta, \text{topo}(\Delta_1) \longrightarrow \Omega, \Omega_1$	Given
$\Omega, \{\Omega_1\}$	
$= \Omega, \Omega_1$	By definition
$= [\Omega, \Omega_1](\Delta, \text{topo}(\Delta_1))$	I.H.
$= [\Omega, \{\Omega_1\}](\Delta, \{\Delta_1\})$	By definition

□

Lemma F.46 (Finishing Completions). *If Ω ok, and $\Omega \longrightarrow \Omega'$, then $[\Omega']\Omega' = \text{topo}([\Omega]\Omega)$.*

PROOF. By induction on $\Omega \longrightarrow \Omega'$. Most cases are the same as Lemma D.34.

For the case

$$\frac{\text{A-CTXE-LO} \quad \Delta \longrightarrow \Theta \quad \Delta, \text{topo}(\Delta_1) \longrightarrow \Theta, \Theta_1}{\Delta, \{\Delta_1\} \longrightarrow \Theta, \{\Theta_1\}}$$

$\Omega, \{\Omega_1\} \longrightarrow \Omega', \{\Omega'_1\} \wedge \Omega, \text{topo}(\Omega_1) \longrightarrow \Omega', \Omega'_1 \wedge \Omega \longrightarrow \Omega'$	Given
$\Omega', \{\Omega'_1\}$	
$= \Omega', \Omega'_1$	By definition
$= \text{topo}(\Omega, \text{topo}(\Omega_1))$	I.H.
$= \text{topo}(\Omega, \Omega_1)$	Follows

□

F.2.4 Decidability.

Lemma F.47 (Promotion Preserves $\langle \Delta \rangle$). *If $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$, then $\langle \Delta \rangle = \langle \Theta \rangle$.*

PROOF. By a straightforward induction on the derivation. □

Lemma F.48 (Unification Makes Progress). *If $\Delta \Vdash^{\mu} \omega_1 \approx \omega_2 \dashv \Theta$, then either $\Theta = \Delta$, or $\langle \Theta \rangle < \langle \Delta \rangle$.*

PROOF. By induction on the derivation.

- In rule **A-U-REFL-TT**, the goal holds trivially.
- Case

$$\frac{\text{A-U-KVARL-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \quad \Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1 \Vdash^{\mu} [\Theta_1] \omega_1 \approx \omega_2 \dashv \Theta_3}{\Delta \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \dashv \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2}$$

$\langle \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \rangle = \langle \Delta \rangle$ $\Theta_3 = \Theta_1 \cup \langle \Theta_3 \rangle < \langle \Theta_1 \rangle$ $\langle \Theta_3 \rangle \leq \langle \Theta_1 \rangle$ $\langle \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2 \rangle < \langle \Theta_3, \widehat{\alpha} : \omega_1, \Theta_2 \rangle$ $\leq \langle \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \rangle$ $= \langle \Delta \rangle$	Lemma F.47 I.H. Follows
--	--------------------------------------

- The case for rule **A-U-KVARR-TT** is similar as the previous case.
- Case

$$\frac{\text{A-U-KVARL-LO-TT} \quad \Delta_1, \Delta_2 \dashv^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \quad \Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2] \omega_1 \approx \omega_2 \dashv \Theta_5, \{\Theta_6\}}{\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \dashv \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4}$$

$\langle \Theta \rangle = \langle \Delta_1, \widehat{\alpha} : \omega_1, \Delta_2 \rangle$ $\langle \Delta[\{\Theta\}] \rangle = \langle \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \rangle$ $\Theta_5, \{\Theta_6\} = \Theta_1, \{\Theta_2\} \cup \Theta_5, \{\Theta_6\} < \langle \Theta_1, \{\Theta_2\} \rangle$ $\langle \Theta_5, \{\Theta_6\} \rangle \leq \langle \Theta_1, \{\Theta_2\} \rangle$ $\langle \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4 \rangle$ $< \langle \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \rangle$ $\leq \langle \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \rangle$ $= \langle \Delta[\{\Theta\}] \rangle$ $= \langle \Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \rangle$	By moving By Lemma F.47 I.H. Follows
--	---

- The case for rule **A-U-KVARR-LO-TT** is similar as the previous case.
- Case

$$\frac{\text{A-U-APP} \quad \Delta \Vdash^{\mu} \rho_1 \approx \rho_3 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\mu} [\Delta_1] \rho_2 \approx [\Delta_1] \rho_4 \dashv \Theta}{\Delta \Vdash^{\mu} \rho_1 \rho_2 \approx \rho_3 \rho_4 \dashv \Theta}$$

$\Delta_1 = \Delta \cup \langle \Delta_1 \rangle < \langle \Delta \rangle$ $\Delta_1 = \Theta \cup \langle \Theta \rangle < \langle \Delta_1 \rangle$ If $\Delta_1 = \Delta$ and $\Delta_1 = \Theta$ $\Delta = \Theta$ Otherwise $\langle \Theta \rangle < \langle \Delta \rangle$	I.H. I.H. Follows directly Follows directly
---	--

- The case for rule **A-U-KAPP** is similar as the previous case. □

Lemma F.49 (Promotion Preserves $|\rho|$). *Given a context $\Delta[\widehat{\alpha}]$ ok, if $\Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$, then for all ρ , we have $|\Delta[\rho]| = |\Theta[\rho]|$.*

PROOF. By a straightforward induction on the promotion judgment.

- Most cases we have $\Delta = \Theta$. So the goal follows trivially.
- Case

$$\frac{\text{A-PR-APP} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash_{\widehat{\alpha}}^{\text{Pr}} [\Delta_1]\omega_2 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \omega_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}}$$

The goal follows directly from I.H..

- The case for rule **A-PR-KAPP** is the same as the previous case.
- Case

$$\frac{\text{A-PR-KUVARR-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} [\Delta]\rho \rightsquigarrow \rho_1 \dashv \Theta[\widehat{\alpha}][\widehat{\beta} : \rho]}{\Delta[\widehat{\alpha}][\widehat{\beta} : \rho] \Vdash_{\widehat{\alpha}}^{\text{Pr}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_1 \dashv \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]}}$$

From I.H., forall ρ' , we have $|\Theta[\widehat{\alpha}][\widehat{\beta} : \rho]|\rho'| = |\Delta[\rho']|$. As compared to $\Theta[\widehat{\alpha}][\widehat{\beta} : \rho]$, $\Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]$ only substituted $\widehat{\beta}$ with $\widehat{\beta}_1$, which preserves the size. Therefore $|\Theta[\widehat{\alpha}][\widehat{\beta} : \rho]|\rho'| = |\Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]|\rho'|$. So $|\Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]|\rho'| = |\Delta[\rho']|$. □

Theorem F.50 (Promotion Terminates). *Given a context $\Delta[\widehat{\alpha}]$ ok, and a kind ρ_1 with $[\Delta]\rho_1 = \rho_1$, it is decidable whether there exists Θ such that $\Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$.*

PROOF. Draw the dependency graph of the input context. We measure the promotion process $\Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$ by the lexicographic order of

- (1) the maximal height of the being promoted types in the dependency graph;
- (2) $|\omega_1|$.

We prove the measurement always get smaller from the conclusion to the hypothesis.

We first prove (1) gets no larger from the conclusion to the premises. This can be done via a straightforward induction on the promotion judgment.

Now we induction on the promotion judgment.

- Most cases do not have hypothesis.
- Case

$$\frac{\text{A-PR-APP} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash_{\widehat{\alpha}}^{\text{Pr}} [\Delta_1]\omega_2 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta \Vdash_{\widehat{\alpha}}^{\text{Pr}} \omega_1 \omega_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}}$$

$ \omega_1 < \omega_1 \omega_2 $ $ \Delta_1]\omega_2 = \Delta]\omega_2 $ $= \omega_2 $ $< \omega_1 \omega_2 $	Follows directly By Lemma F.49 Given the equation Follows
--	--

- The case for rule **A-PR-KAPP** is the same as the previous case.

- Case

$$\frac{\text{A-PR-KUVR-R-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} [\Delta]\rho \rightsquigarrow \rho_1 \dashv \Theta[\widehat{\alpha}][\widehat{\beta} : \rho]}{\Delta[\widehat{\alpha}][\widehat{\beta} : \rho] \Vdash_{\widehat{\alpha}}^{\text{pr}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_1 \dashv \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]}$$

In the dependency graph, there are edges from $\widehat{\beta}$ to $[\Delta]\rho$. So the height gets decreased from the conclusion to the hypothesis. □

Theorem F.51 (Unification Terminates). *Given a context Δ ok, and kinds ρ_1 and ρ_2 , where $[\Delta]\rho_1 = \rho_1$, and $[\Delta]\rho_2 = \rho_2$, it is decidable whether there exists Θ such that $\Delta \Vdash^{\mu} \rho_1 \approx \rho_2 \dashv \Theta$.*

PROOF. We measure the unification derivation by the lexicographic order on:

- (1) $\langle \Delta \rangle$
- (2) $|\rho_1|$

We case analyze the derivation.

- The case for rule **A-U-REFL-TT** is decidable.
- Case

$$\frac{\text{A-U-KVARL-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \quad \Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1 \Vdash^{\mu} [\Theta_1]\omega_1 \approx \omega_2 \dashv \Theta_3}{\Delta \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \dashv \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2}$$

$$\left. \begin{array}{l} \langle \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \rangle = \langle \Delta \rangle \\ \langle \Theta_1 \rangle < \langle \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \rangle = \langle \Delta \rangle \end{array} \right\} \begin{array}{l} \text{Lemma F.47} \\ \text{Follows} \end{array}$$

- The case for rule **A-U-KVAR-R-TT** is similar as the previous case.
- Case

$$\frac{\text{A-U-KVARL-LO-TT} \quad \Delta_1, \Delta_2 \dashv^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \quad \Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2]\omega_1 \approx \omega_2 \dashv \Theta_5, \{\Theta_6\}}{\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \dashv \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4}$$

$$\left. \begin{array}{l} \langle \Theta \rangle = \langle \Delta_1, \widehat{\alpha} : \omega_1, \Delta_2 \rangle \\ \langle \Delta[\{\Theta\}] \rangle = \langle \Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \rangle \\ \langle \Delta[\{\Theta\}] \rangle = \langle \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \rangle \\ \langle \Theta_1, \{\Theta_2\} \rangle \\ < \langle \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \rangle \\ = \langle \Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \rangle \end{array} \right\} \begin{array}{l} \text{By moving} \\ \text{Follows} \\ \text{Lemma F.47} \end{array}$$

- The case for rule **A-U-KVAR-R-LO-TT** is similar as the previous case.
- Case

$$\frac{\text{A-U-APP} \quad \Delta \Vdash^{\mu} \rho_1 \approx \rho_3 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\mu} [\Delta_1]\rho_2 \approx [\Delta_1]\rho_4 \dashv \Theta}{\Delta \Vdash^{\mu} \rho_1 \rho_2 \approx \rho_3 \rho_4 \dashv \Theta}$$

For the first condition, we know that $\langle \Delta \rangle = \langle \Delta \rangle$ and the size of the expression decreases.

For the second condition, from Lemma F.48, we know that either $\Delta_1 = \Delta$, or $\langle \Delta_1 \rangle < \langle \Delta \rangle$. In the former case, we know that $[\Delta_1]\rho_2 = \rho_2$. So the size of the expression decreases. In the latter case, we have $\langle \Delta_1 \rangle < \langle \Delta \rangle$ so we are done.

- The case for rule **A-U-KAPP** is similar as the previous one.

□

F.2.5 Source of Unification Variables.

Lemma F.52 (Source of Unification Variables). *If $\Delta \Vdash^k \sigma : \eta \rightsquigarrow \mu \vdash \Theta$, then for any $\widehat{\alpha} \in \text{unsolved}(\Theta)$, either $\widehat{\alpha} \in \text{fkv}([\Theta]\mu)$, or there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$ such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$.*

PROOF. This lemma depends on the similar lemma on many judgments, including kind checking, instantiation, and unification. We prove them one by one.

When the input context is the same as the output context, the lemma holds trivially, as all unsolved unification variables in Θ are in Δ . So we will skip the discussion of those cases.

Part 1: Kinding By induction on the judgment.

- Case

A-KTT-FORALL

$$\Delta \Vdash^k \kappa \Leftarrow \star \rightsquigarrow \omega \vdash \Delta_1 \quad \Delta_1, a : \omega \Vdash^k \sigma \Leftarrow \star \rightsquigarrow \mu \vdash \Delta_2, a : \omega, \Delta_3 \quad \Delta_3 \hookrightarrow a$$

$$\hline \Delta \Vdash^k \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega. [\Delta_3]\mu \vdash \Delta_2, \text{unsolved}(\Delta_3)$$

Given $\widehat{\alpha} \in \text{unsolved}(\Delta_2, \text{unsolved}(\Delta_3))$, we know that $\widehat{\alpha} \in \text{unsolved}(\Delta_2, a : \omega, \Delta_3)$.

Then by the lemma on kind checking. we have two cases.

(1) $\widehat{\alpha} \in \text{fkv}([\Delta_2, a : \omega, \Delta_3]\mu)$. Then

(a) $\widehat{\alpha} \in \text{fkv}(\mu)$, and $\widehat{\alpha}$ is unsolved in $\Delta_2, a : \omega, \Delta_3$.

Therefore $\widehat{\alpha} \in \text{fkv}([\Delta_3]\mu)$.

Since $\widehat{\alpha} \in \text{unsolved}(\Delta_2, \text{unsolved}(\Delta_3))$, we have $\widehat{\alpha} \in \text{fkv}([\Delta_2, \text{unsolved}(\Delta_3)]([\Delta_3]\mu))$ so we are done.

(b) there exists a $\widehat{\beta}_2 \in \text{fkv}(\mu)$, such that $\widehat{\alpha} \in \text{fkv}([\Delta_2, a : \omega, \Delta_3]\widehat{\beta}_2)$.

Now the goal is to prove $\widehat{\alpha} \in \text{fkv}([\Delta_2, \text{unsolved}(\Delta_3)]([\Delta_3]\widehat{\beta}_2))$.

Notice that $[\Delta_2, \text{unsolved}(\Delta_3)]([\Delta_3]\widehat{\beta}_2) = [\Delta_2]([\Delta_3]\widehat{\beta}_2) = [\Delta_2, a : \omega, \Delta_3]\widehat{\beta}_2$.

So we are done.

(2) there exists $\widehat{\beta}_1 \in \text{unsolved}(\Delta_1, a : \omega)$ such that $\widehat{\alpha} \in \text{fkv}([\Delta_2, a : \omega, \Delta_3]\widehat{\beta}_1)$.

Because $\widehat{\beta}_1$ is in $\Delta_1, a : \omega$, then it must be $\widehat{\beta}_1$ in $\Delta_2, a : \omega$ by Lemma F.29 and Lemma F.21.

Therefore $[\Delta_2, a : \omega, \Delta_3]\widehat{\beta}_1 = [\Delta_2]\widehat{\beta}_1$.

So we have $\widehat{\alpha} \in \text{fkv}([\Delta_2]\widehat{\beta}_1)$.

Also, it must be $\widehat{\beta}_1 \in \text{unsolved}(\Delta_1)$. Then by the lemma on kind checking. we have two subcases.

(a) $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\omega)$.

We know that $\Delta_1 \longrightarrow \Delta_2$ by Lemma F.15 and Lemma F.29.

So $[\Delta_2, \text{unsolved}(\Delta_3)]\omega = [\Delta_2]\omega = [\Delta_2]([\Delta_1]\omega)$.

We already know that $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\omega)$ and $\widehat{\alpha} \in \text{fkv}([\Delta_2]\widehat{\beta}_1)$, so we know $\widehat{\alpha} \in \text{fkv}([\Delta_2]([\Delta_1]\omega))$ and we are done.

(b) there exists $\widehat{\beta}_3 \in \text{unsolved}(\Delta)$ such that $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\widehat{\beta}_3)$.

Similar as the previous subcase, we have $[\Delta_2, \text{unsolved}(\Delta_3)]\widehat{\beta}_3 = [\Delta_2]\widehat{\beta}_3 = [\Delta_2]([\Delta_1]\widehat{\beta}_3)$.

We already know that $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\widehat{\beta}_3)$ and $\widehat{\alpha} \in \text{fkv}([\Delta_2]\widehat{\beta}_1)$, so we know $\widehat{\alpha} \in \text{fkv}([\Delta_2]([\Delta_1]\widehat{\beta}_3))$ and we are done.

- The case for rule A-KTT-FORALLI is similar as the previous case.

- Case

A-KTT-APP

$$\Delta \Vdash^k \tau_1 : \eta_1 \rightsquigarrow \rho_1 \vdash \Delta_1 \quad \Delta_1 \Vdash^{\text{kapp}} (\rho_1 : [\Delta_1]\eta_1) \bullet \tau_2 : \omega \rightsquigarrow \rho \vdash \Theta$$

$$\hline \Delta \Vdash^k \tau_1 \tau_2 : \omega \rightsquigarrow \rho \vdash \Theta$$

Given $\widehat{\alpha} \in \text{unsolved}(\Theta)$, by the lemma on application kinding part we have two cases.

(1) $\widehat{\alpha} \in \text{fkv}([\Theta]\rho)$. Then the goal follows directly.

(2) there exists $\widehat{\beta}_1 \in \text{unsolved}(\Delta_1)$ such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta}_1)$.

Because $\widehat{\beta}_1 \in \text{unsolved}(\Delta_1)$, by I.H., we have two subcases.

(a) $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\rho_1)$.

Then by Lemma F.30 we have $[\Theta]\rho_1 = [\Theta]([\Delta_1]\rho_1)$.

We already know that $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\rho_1)$ and $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta}_1)$ so we must have $\widehat{\alpha} \in \text{fkv}([\Theta]\rho_1)$.

By the lemma on application kinding, we have $\widehat{\alpha} \in \text{fkv}([\Theta]\rho)$, so we are done.

(b) there exists $\widehat{\beta}_2 \in \text{unsolved}(\Delta_1)$, such that $\widehat{\beta}_1 \in \text{fkv}([\Delta_1]\widehat{\beta}_2)$

By Lemma F.30 we have $[\Theta]\widehat{\beta}_2 = [\Theta]([\Delta_1]\widehat{\beta}_2)$.

And we must have $\widehat{\alpha} \in \text{fkv}([\Theta]([\Delta_1]\widehat{\beta}_2))$.

- The case for rules **A-KTT-KAPP** and **A-KTT-KAPP-INFER** is similar as the previous case.

Instantiation The statement for instantiation is: if $\Delta \Vdash^{\text{inst}} \mu_1 : \eta_1 \sqsubseteq \eta_2 \rightsquigarrow \mu_2 + \Theta$, then for any $\widehat{\alpha} \in \text{unsolved}(\Theta)$, either $\widehat{\alpha} \in \text{fkv}([\Theta]\mu_2)$, or there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$, such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$. Moreover, μ_2 contains all the unification variables in μ_1 .

We prove it by induction on the derivation.

- Case

$$\frac{\text{A-INST-REFL} \quad \Delta \Vdash^{\text{u}} \omega_1 \approx \omega_2 + \Theta}{\Delta \Vdash^{\text{inst}} \mu : \omega_1 \sqsubseteq \omega_2 \rightsquigarrow \mu + \Theta}$$

The first half of the goal follows directly from the lemma on unification part, and the second goal holds trivially.

- Case

$$\frac{\text{A-INST-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{inst}} \mu_1 @\widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}] \sqsubseteq \omega_2 \rightsquigarrow \mu_2 + \Theta}{\Delta \Vdash^{\text{inst}} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2 + \Theta}$$

The second half of the goal follows directly from I.H.. Given $\widehat{\alpha}_1 \in \text{unsolved}(\Theta)$, by I.H., we have two cases.

(1) $\widehat{\alpha}_1 \in \text{fkv}([\Theta]\mu_2)$. So the first half of the goal holds directly.

(2) there exists $\widehat{\beta} \in \text{unsolved}(\Delta, \widehat{\alpha} : \omega_1)$, such that $\widehat{\alpha}_1 \in \text{fkv}([\Theta]\widehat{\beta})$.

Then we have either $\widehat{\beta} = \widehat{\alpha}$, or $\widehat{\beta} \in \text{unsolved}(\Delta)$. In the former case, as $\mu_1 @\widehat{\alpha}$ contains $\widehat{\alpha}$, we have μ_2 contains $\widehat{\alpha}$. Therefore $\widehat{\alpha}_1 \in \text{fkv}([\Theta]\mu_2)$ and we are done. In the latter case, the goal follows directly.

- The case for rule **A-INST-FORALL-INFER** is similar as the previous case.

Application Kinding The statement for application kinding is: if $\Delta \Vdash^{\text{kapp}} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 + \Theta$, then for any $\widehat{\alpha} \in \text{unsolved}(\Theta)$, either $\widehat{\alpha} \in \text{fkv}([\Theta]\rho_2)$, or there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$, such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$. Moreover, ρ_2 contains all the unification variables in ρ_1 .

We prove it by induction on the derivation.

- Case

$$\frac{\text{A-KAPP-TT-ARROW} \quad \Delta \Vdash^{\text{kc}} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_2 + \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \omega_1 \rightarrow \omega_2) \bullet \tau : \omega_2 \rightsquigarrow \rho_1 \rho_2 + \Theta}$$

The first half of the goal follows directly from the lemma on kind checking part.

The second half of the goal holds trivially.

- Case

$$\frac{\text{A-KAPP-TT-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{kapp}} (\rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}]) \bullet \tau : \omega \rightsquigarrow \rho \vdash \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \forall a : \omega_1. \eta) \bullet \tau : \omega \rightsquigarrow \rho \vdash \Theta}$$

The second half of the goal follows directly from I.H..

Given $\widehat{\alpha}_1 \in \text{unsolved}(\Theta)$, by I.H., we have two cases.

- (1) $\widehat{\alpha}_1 \in \text{fkv}([\Theta]\rho)$. So the first half of the goal holds directly.
- (2) there exists $\widehat{\beta} \in \text{unsolved}(\Delta, \widehat{\alpha} : \omega_1)$, such that $\widehat{\alpha}_1 \in \text{fkv}([\Theta]\widehat{\beta})$.

Then we have either $\widehat{\beta} = \widehat{\alpha}$, or $\widehat{\beta} \in \text{unsolved}(\Delta)$. In the former case, as $\rho_1 @ \widehat{\alpha}$ contains $\widehat{\alpha}$, we have ρ contains $\widehat{\alpha}$. Therefore $\widehat{\alpha}_1 \in \text{fkv}([\Theta]\rho)$ and we are done. In the latter case, the goal follows directly.

- The case for rule **A-KAPP-TT-FORALL-INFER** is the same as previous case.
- Case

$$\frac{\text{A-KAPP-TT-KUVAR} \quad \Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \Vdash^{\text{kcc}} \tau \leftarrow \widehat{\alpha}_1 \rightsquigarrow \rho_2 \vdash \Theta}{\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash^{\text{kapp}} (\rho_1 : \widehat{\alpha}) \bullet \tau : \widehat{\alpha}_2 \rightsquigarrow \rho_1 \rho_2 \vdash \Theta}$$

The second half of the goal follows trivially.

Given $\widehat{\alpha}_3 \in \text{unsolved}(\Theta)$, by I.H., we have two cases.

- (1) $\widehat{\alpha}_3 \in \text{fkv}([\Theta]\rho_2)$. So the first half of the goal holds directly.
- (2) there exists $\widehat{\beta} \in \text{unsolved}(\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2)$, such that $\widehat{\alpha}_3 \in \text{fkv}([\Theta]\widehat{\beta})$.

Then we have either $\widehat{\beta} = \widehat{\alpha}_1$, or $\widehat{\beta} = \widehat{\alpha}_2$, or $\widehat{\beta} \in \text{unsolved}(\Delta_1, \widehat{\alpha} : \omega = \widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2, \Delta_2)$. In the former two cases, we pick $\widehat{\alpha}$ from the input context. And $[\Theta]\widehat{\alpha} = [\Theta](\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \kappa = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2)\widehat{\alpha} = [\Theta](\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2)$ by Lemma F.30. Therefore $\widehat{\alpha}_3 \in \text{fkv}([\Theta]\widehat{\alpha})$.

In the later case, then it must be $\widehat{\beta} \in \text{unsolved}(\Delta_1, \widehat{\alpha} : \omega, \Delta_2)$ So we are done.

Kind Checking The statement for kind checking is: if $\Delta \Vdash^{\text{kcc}} \sigma \leftarrow \eta \rightsquigarrow \mu \vdash \Theta$, then for any $\widehat{\alpha} \in \text{unsolved}(\Theta)$, either $\widehat{\alpha} \in \text{fkv}([\Theta]\mu)$, or there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$, such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$.

To prove the lemma, we have

$$\frac{\text{A-KC-SUB} \quad \Delta \Vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu_1 \vdash \Delta_1 \quad \Delta_1 \Vdash^{\text{inst}} \mu_1 : [\Delta_1]\eta \sqsubseteq [\Delta_1]\omega \rightsquigarrow \mu_2 \vdash \Delta_2}{\Delta \Vdash^{\text{kcc}} \sigma \leftarrow \omega \rightsquigarrow \mu_2 \vdash \Delta_2}$$

Given $\widehat{\alpha} \in \text{unsolved}(\Delta_2)$, by the lemma on the instantiation part, we have two cases.

- (1) $\widehat{\alpha} \in \text{fkv}([\Delta_2]\mu_2)$. Then the goal follows directly.
- (2) there exists $\widehat{\beta} \in \text{unsolved}(\Delta_1)$, such that $\widehat{\alpha} \in \text{fkv}([\Delta_2]\widehat{\beta})$. Then because $\widehat{\beta} \in \text{unsolved}(\Delta_1)$, by the lemma on the kinding part, we have two subcases.
 - (a) $\widehat{\beta} \in \text{fkv}([\Delta_1]\mu_1)$. Then by the lemma on the instantiation part, we know that $\widehat{\beta} \in \text{fkv}([\Delta_1]\mu_2)$. By Lemma F.30, we have $[\Delta_2]\mu_2 = [\Delta_2]([\Delta_1]\mu_2)$. So we have $\widehat{\alpha} \in \text{fkv}([\Delta_2]([\Delta_1]\mu_2))$.
 - (2) there exists $\widehat{\beta}_2 \in \text{unsolved}(\Delta)$, such that $\widehat{\beta} \in \text{fkv}([\Delta_1]\widehat{\beta}_2)$. By Lemma F.30, we have $[\Delta_2]\widehat{\beta}_2 = [\Delta_2]([\Delta_1](\widehat{\beta}_2))$. So we have $\widehat{\alpha} \in \text{fkv}([\Delta_2]([\Delta_1]\widehat{\beta}_2))$.

Promotion The statement for promotion is: if $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \vdash \Theta$, then for any $\widehat{\alpha}' \in \text{unsolved}(\Theta)$, there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$, such that $\widehat{\alpha}' \in \text{fkv}([\Theta]\widehat{\beta})$.

The only interesting case here is

$$\frac{\text{A-PR-KUVRAR-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{PR}} [\Delta]\rho \rightsquigarrow \rho_1 \div \Theta[\widehat{\alpha}][\widehat{\beta} : \rho]}{\Delta[\widehat{\alpha}][\widehat{\beta} : \rho] \Vdash_{\widehat{\alpha}}^{\text{PR}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_1 \div \Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]}$$

Given $\widehat{\alpha}' \in \text{unsolved}(\Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1])$, we have two cases:

- $\widehat{\alpha}'$ is not $\widehat{\beta}_1$.

Then we have $\widehat{\alpha}' \in \text{unsolved}(\Theta[\widehat{\alpha}][\widehat{\beta} : \rho])$, and by I.H. we are done.

- $\widehat{\alpha}'$ is $\widehat{\beta}_1$.

Then we pick $\widehat{\beta}$ from the input context, and we have that $[\Theta[\widehat{\beta}_1 : \rho_1, \widehat{\alpha}][\widehat{\beta} : \rho = \widehat{\beta}_1]]\widehat{\beta} = \widehat{\beta}_1$ so we are done.

Unification The statement for unification is: if $\Delta \Vdash^{\mu} \omega_1 \approx \omega_2 \div \Theta$, then for any $\widehat{\alpha} \in \text{unsolved}(\Theta)$, there exists $\widehat{\beta} \in \text{unsolved}(\Delta)$, such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$.

Here, all cases are essentially the same. We discuss two of them and the rest can be proved in a similar way.

- Case

$$\frac{\text{A-U-APP} \quad \Delta \Vdash^{\mu} \rho_1 \approx \rho_3 \div \Delta_1 \quad \Delta_1 \Vdash^{\mu} [\Delta_1]\rho_2 \approx [\Delta_1]\rho_4 \div \Theta}{\Delta \Vdash^{\mu} \rho_1 \rho_2 \approx \rho_3 \rho_4 \div \Theta}$$

Given $\widehat{\alpha} \in \text{unsolved}(\Theta)$, by I.H., we know that there exists $\widehat{\beta} \in \text{unsolved}(\Delta_1)$, such that $\widehat{\alpha} \in \text{fkv}([\Theta]\widehat{\beta})$.

And because $\widehat{\beta} \in \text{unsolved}(\Delta_1)$, by I.H., we know that there exists $\widehat{\beta}_2 \in \text{unsolved}(\Delta)$, such that $\widehat{\beta} \in \text{fkv}([\Delta_1]\widehat{\beta}_2)$.

By Lemma F.30 we know that $[\Theta]\widehat{\beta}_2 = [\Theta]([\Delta_1]\widehat{\beta}_2)$. So we must have $\widehat{\alpha} \in \text{fkv}([\Theta]([\Delta_1]\widehat{\beta}_2))$.

- Case

$$\frac{\text{A-U-KVARL-LO-TT} \quad \Delta_1, \Delta_2 \Vdash^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{PR}} \rho_1 \rightsquigarrow \rho_2 \div \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4}{\Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2]\omega_1 \approx \omega_2 \div \Theta_5, \{\Theta_6\}} \quad \Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \div \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4}$$

Given $\widehat{\alpha}_1 \in \text{unsolved}(\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4)$, we have two cases to discuss.

- (1) $\widehat{\alpha}_1 \in \text{unsolved}(\Theta_5, \{\Theta_6\})$.

Then by I.H., we know that there is $\widehat{\beta}_1 \in \text{unsolved}(\Theta_1, \{\Theta_2\})$ such that $\widehat{\alpha}_1 \in \text{fkv}([\Theta_5, \{\Theta_6\}]\widehat{\beta}_1)$.

By the definition, we know that $\widehat{\beta}_1 \in \text{unsolved}(\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4)$.

Then by the lemma on the promotion part, we know that there exists a $\widehat{\beta}_2 \in \text{unsolved}(\Delta[\{\Theta\}])$ such that $\widehat{\beta}_1 \in \text{fkv}([\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4]\widehat{\beta}_2)$.

By the definition of moving, we know that all unsolved unification in $\text{unsolved}(\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2)$ are in $\text{unsolved}(\Theta)$. Therefore we have $\widehat{\beta}_2 \in \text{unsolved}(\Delta[\{\Theta\}])$.

We have that $[\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\widehat{\beta}_2 = [\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]([\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4]\widehat{\beta}_2)$ by Lemma F.30, as $\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$, whose derivation can be found in the proof of Lemma F.10.

Then we must have $\widehat{\alpha}_1 \in \text{fkv}([\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]([\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4]\widehat{\beta}_2))$.

- (2) $\widehat{\alpha}_1$ is in the domain of Θ_3 and Θ_4 .

Then it must be in $\text{unsolved}(\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4)$.

Then by the lemma on the promotion part, we know that there exists a $\widehat{\beta} \in \text{unsolved}(\Delta[\{\Theta\}])$ such that $\widehat{\alpha}_1 \in \text{fkv}([\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4]\widehat{\beta})$.

By moving, we know that all unsolved unification in $\text{unsolved}(\Delta[\{\Theta\}])$ are in $\text{unsolved}(\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}])$.

Therefore we have $\widehat{\beta} \in \text{unsolved}(\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}])$.

We have that $[\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\widehat{\beta} = [\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4](\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4)\widehat{\beta}$ by Lemma F.30, as $\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$, whose derivation can be found in the proof of Lemma F.10.

Then we must have $\widehat{\alpha}_1 \in \text{fkv}([\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4](\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4)\widehat{\beta})$. \square

F.2.6 Soundness of Algorithm.

Lemma F.53 (Soundness of Promotion). *If Δ ok, and $[\Delta]\omega_1 = \omega_1$, and $\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \omega_1 \rightsquigarrow \omega_2 \dashv \Theta$, then $[\Theta]\omega_1 = [\Theta]\omega_2 = \omega_2$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\omega_1 = [\Omega]\omega_2$.*

PROOF. The first half follows directly from a straightforward induction on promotion.

The second half of the goal follows directly from and Lemma F.30. \square

Lemma F.54 (Soundness of Unification). *If Δ ok, and $\Delta \Vdash^{\mu} \omega_1 \approx \omega_2 \dashv \Theta$, then $[\Theta]\omega_1 = [\Theta]\omega_2$. If $\Theta \longrightarrow \Omega$, then $[\Omega]\omega_1 = [\Omega]\omega_2$.*

PROOF. By Lemma F.30, we only need to prove the first half of the lemma.

The case for rule **A-U-REFL-TT** holds trivially. And the case for rule **A-U-APP** and rule **A-U-KAPP** follows from I.H. and Lemma F.30. As rule **A-U-KVARL-TT** and rule **A-U-KVARR-TT**, rule **A-U-KVARL-LO-TT** and rule **A-U-KVARR-LO-TT** are symmetric, we only prove one of them.

- Case

$$\frac{\text{A-U-KVARL-TT} \quad \Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \quad \Theta_1 \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1 \Vdash^{\mu} [\Theta_1]\omega_1 \approx \omega_2 \dashv \Theta_3}{\Delta \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \dashv \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2}$$

$\Theta_1, \widehat{\alpha} : \omega_1, \Theta_2 \longrightarrow \Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2$ $[\Theta_1, \widehat{\alpha} : \omega_1, \Theta_2]\rho_1 = [\Theta_1, \widehat{\alpha} : \omega_1, \Theta_2]\rho_2$ $[\Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2]\rho_1 = [\Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2]\rho_2$ $[\Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2]\widehat{\alpha} = [\Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2]\rho_2$ $[\Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2]\widehat{\alpha} = [\Theta_3, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_2]\rho_1$	We have proved in Lemma F.10 By Lemma F.53 By Lemma F.30 By definition By equations
--	---

- Case

$$\frac{\text{A-U-KVARL-LO-TT} \quad \Delta_1, \Delta_2 \dashv^{\text{mv}} \widehat{\alpha} : \omega_1 \rightsquigarrow \Theta \quad \Delta[\{\Theta\}] \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_1 \rightsquigarrow \rho_2 \dashv \Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \quad \Theta_1, \{\Theta_2\} \Vdash^{\text{ela}} \rho_2 : \omega_2 \quad \Theta_1, \{\Theta_2\} \Vdash^{\mu} [\Theta_1, \Theta_2]\omega_1 \approx \omega_2 \dashv \Theta_5, \{\Theta_6\}}{\Delta[\{\Delta_1, \widehat{\alpha} : \omega_1, \Delta_2\}] \Vdash^{\mu} \widehat{\alpha} \approx \rho_1 \dashv \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4}$$

$\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4 \longrightarrow \Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4$ $[\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4]\rho_1 = [\Theta_1, \{\Theta_2, \widehat{\alpha} : \omega_1, \Theta_3\}, \Theta_4]\rho_2$ $[\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\rho_1 = [\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\rho_2$ $[\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\widehat{\alpha} = [\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\rho_2$ $[\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\rho_1 = [\Theta_5, \{\Theta_6, \widehat{\alpha} : \omega_1 = \rho_2, \Theta_3\}, \Theta_4]\widehat{\alpha}$	We have proved in Lemma F.10 By Lemma F.53 By Lemma F.30 By definition By equations
--	---

\square

Lemma F.55 (Soundness of Instantiation). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu_1 : \eta$, and $\Delta \Vdash^{\text{ela}} \omega : \star$, and $\Delta \Vdash^{\text{inst}} \mu_1 : \eta \sqsubseteq \omega \rightsquigarrow \mu_2 \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\mu_1 : [\Omega]\eta \sqsubseteq [\Omega]\omega \rightsquigarrow [\Omega]\mu_2$.*

- Case

$$\frac{\text{A-INST-REFL} \quad \Delta \Vdash^{\text{u}} \omega_1 \approx \omega_2 \dashv \Theta}{\Delta \Vdash^{\text{inst}} \mu : \omega_1 \sqsubseteq \omega_2 \rightsquigarrow \mu \dashv \Theta}$$

$$\begin{array}{l|l} [\Omega]\omega_1 = [\Omega]\omega_2 & \text{By Lemma F.54} \\ [\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\mu : [\Omega]\omega_1 \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu & \text{By rule INST-REFL} \end{array}$$

- Case

$$\frac{\text{A-INST-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{inst}} \mu_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}] \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}{\Delta \Vdash^{\text{inst}} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2 \dashv \Theta}$$

$\Theta \longrightarrow \Omega$ $[\Omega](\Delta, \widehat{\alpha} : \omega_1) \Vdash^{\text{inst}} [\Omega]\mu_1 @ ([\Omega]\widehat{\alpha}) : ([\Omega](\eta[a \mapsto \widehat{\alpha}]) \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu_2$ $[\Omega](\Delta, \widehat{\alpha} : \omega_1) \Vdash^{\text{inst}} [\Omega]\mu_1 @ ([\Omega]\widehat{\alpha}) : ([\Omega]\eta)[a \mapsto [\Omega]\widehat{\alpha}] \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu_2$ $\Delta \Vdash^{\text{ela}} \mu_1 : \forall a : \omega_1. \eta$ $\Delta \Vdash^{\text{ela}} \omega_1 : \star$ $\Delta \longrightarrow \Delta, \widehat{\alpha} : \omega_1$ $\Delta, \widehat{\alpha} : \omega_1 \longrightarrow \Theta$ $\Theta \longrightarrow \Omega$ $\Delta, \widehat{\alpha} : \omega_1 \longrightarrow \Omega \wedge \Delta \longrightarrow \Omega$ $[\Omega](\Delta, \widehat{\alpha} : \omega_1) = [\Omega]\Delta$ $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\mu_1 @ ([\Omega]\widehat{\alpha}) : ([\Omega]\eta)[a \mapsto [\Omega]\widehat{\alpha}] \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu_2$ $\Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{ela}} \widehat{\alpha} : [\Delta]\omega_1$ $[\Omega](\Delta, \widehat{\alpha} : \omega_1) \Vdash^{\text{ela}} [\Omega]\widehat{\alpha} : [\Omega]([\Delta]\omega_1)$ $[\Omega]\Delta \Vdash^{\text{ela}} [\Omega]\widehat{\alpha} : [\Omega]([\Delta]\omega_1)$ $[\Omega]\Delta \Vdash^{\text{ela}} [\Omega]\widehat{\alpha} : [\Omega]\omega_1$ $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\mu_1 : \forall a : [\Omega]\omega_1. [\Omega]\eta \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu_2$	Given I.H. By substitution Given By inversion By rule -A-CTXE-ADD-TT By Lemma F.11 Given By Lemma F.32 By Lemma F.45 By equation By rule A-ELA-KUVAR By Lemma F.57 By equation By Lemma F.30 By rule INST-FORALL
---	---

- The case for rule A-INST-FORALLI is similar as the previous one.

□

Lemma F.56 (Soundness of Kinding). *If Δ ok, we have*

- if $\Delta \Vdash^{\text{k}} \sigma : \eta \rightsquigarrow \mu \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{k}} \sigma : [\Omega]\eta \rightsquigarrow [\Omega]\mu$;
- if $\Delta \Vdash^{\text{k}} \sigma \Leftarrow \eta \rightsquigarrow \mu \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{k}} \sigma \Leftarrow [\Omega]\eta \rightsquigarrow [\Omega]\mu$.
- if $\Delta \Vdash^{\text{kapp}} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 \dashv \Theta$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \eta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\rho_1 : [\Omega]\eta \sqsubseteq (\omega_1 \longrightarrow [\Omega]\omega) \rightsquigarrow \rho_3$, and $[\Omega]\Delta \Vdash^{\text{k}} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_4$. and $[\Omega]\rho_2 = \rho_3 \rho_4$.

PROOF. By induction on the derivation.

Part 1 • The case for rules A-KTT-STAR, A-KTT-NAT, A-KTT-VAR, A-KTT-TCON, and A-KTT-ARROW are straightforward.

- Case

$$\frac{\text{A-KTT-APP} \quad \Delta \Vdash^{\text{k}} \tau_1 : \eta_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\text{kapp}} (\rho_1 : [\Delta_1]\eta_1) \bullet \tau_2 : \omega \rightsquigarrow \rho \dashv \Theta}{\Delta \Vdash^{\text{k}} \tau_1 \tau_2 : \omega \rightsquigarrow \rho \dashv \Theta}$$

By well-formedness of the judgments, we know every output context is an extension of the input context and by transitivity we have that output context is an extension of all the previous input contexts.

$ \begin{array}{l} [\Omega]\Delta \Vdash^k \tau_1 : [\Omega]\eta_1 \rightsquigarrow [\Omega]\rho_1 \\ \Delta_1 \Vdash^{\text{ela}} \rho_1 : [\Delta_1]\eta_1 \\ [\Omega]\Delta_1 \Vdash^{\text{inst}} [\Omega]\rho_1 : [\Omega]([\Delta_1]\eta_1) \sqsubseteq \omega_1 \rightarrow [\Omega]\omega \rightsquigarrow \rho_2 \\ \wedge [\Omega]\Delta_1 \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega_1 \rightsquigarrow \rho_3 \\ \wedge [\Omega]\rho = \rho_2 \rho_3 \\ [\Omega]\Delta = [\Omega]\Delta_1 \\ [\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\rho_1 : [\Omega]\eta_1 \sqsubseteq \omega_1 \rightarrow [\Omega]\omega \rightsquigarrow \rho_2 \\ [\Omega]\Delta \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega_1 \rightsquigarrow \rho_3 \\ [\Omega]\Delta \Vdash^k \tau_1 \tau_2 : [\Omega]\omega \rightsquigarrow [\Omega]\rho \end{array} $	<p>I.H. By Lemma F.15 By Part 3</p> <p>By Lemma F.45 By equations and Lemma F.30 By equations By rule KTT-APP</p>
---	---

- Case

$$\frac{\text{A-KTT-KAPP} \quad \Delta \Vdash^k \tau_1 : \eta \rightsquigarrow \rho_1 \dashv \Delta_1 \quad [\Delta_1]\eta = \forall a : \omega. \eta_2 \quad \Delta_1 \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega \rightsquigarrow \rho_2 \dashv \Delta_2}{\Delta \Vdash^k \tau_1 @ \tau_2 : \eta_2[a \mapsto \rho_2] \rightsquigarrow \rho_1 @ \rho_2 \dashv \Delta_2}$$

By well-formedness of the judgments, we know every output context is an extension of the input context and by transitivity we have that output context is an extension of all the previous input contexts.

$ \begin{array}{l} [\Omega]\Delta \Vdash^k \tau_1 : [\Omega]\eta \rightsquigarrow [\Omega]\rho_1 \\ [\Omega]\eta = [\Omega]([\Delta_1]\eta) = \forall a : [\Omega]\omega. [\Omega]\eta_2. \\ [\Omega]\Delta \Vdash^k \tau_1 : \forall a : [\Omega]\omega. [\Omega]\eta_2 \rightsquigarrow [\Omega]\rho_1 \\ [\Omega]\Delta_1 \Vdash^{\text{kc}} \tau_2 \Leftarrow [\Omega]\omega \rightsquigarrow [\Omega]\rho_2 \\ [\Omega]\Delta \Vdash^{\text{kc}} \tau_2 \Leftarrow [\Omega]\omega \rightsquigarrow [\Omega]\rho_2 \\ [\Omega]\Delta \Vdash^k \tau_1 @ \tau_2 : ([\Omega]\eta_2)[a \mapsto [\Omega]\rho_2] \rightsquigarrow [\Omega]\rho_1 @ [\Omega]\rho_2 \\ [\Omega]\Delta \Vdash^k \tau_1 @ \tau_2 : [\Omega](\eta_2[a \mapsto \rho_2]) \rightsquigarrow [\Omega](\rho_1 @ \rho_2) \end{array} $	<p>I.H. By Lemma F.30 By equations By Part 2 By Lemma F.45 By rule KTT-KAPP By substitutions</p>
--	---

- The case for rule **A-KTT-KAPP-INFER** is similar to the previous case.

- Case

$$\frac{\text{A-KTT-FORALL} \quad \Delta \Vdash^{\text{kc}} \kappa \Leftarrow \star \rightsquigarrow \omega \dashv \Delta_1 \quad \Delta_1, a : \omega \Vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow \mu \dashv \Delta_2, a : \omega, \Delta_3 \quad \Delta_3 \hookrightarrow a}{\Delta \Vdash^k \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega. [\Delta_3]\mu \dashv \Delta_2, \text{unsolved}(\Delta_3)}$$

$ \begin{array}{l} \Delta_1, a : \omega \longrightarrow \Delta_2, a : \omega, \Delta_3 \\ \Delta_1 \longrightarrow \Delta_2 \wedge \Delta_3 \text{ soft} \\ \Delta_2 \longrightarrow \Delta_2, \text{unsolved}(\Delta_3) \\ \Delta_2, \text{unsolved}(\Delta_3) \longrightarrow \Omega \\ \Delta_1 \longrightarrow \Omega \\ [\Omega]\Delta \Vdash^{\text{kc}} \kappa \Leftarrow \star \rightsquigarrow [\Omega]\omega \\ \Delta_2 \longrightarrow \Omega_1 \wedge \Omega = \Omega_1, \Omega_2 \wedge \Omega_2 \text{ soft} \\ \Delta_2, a : \omega \longrightarrow \Omega_1, a : \omega \\ \text{Construct a } \Omega_3 \text{ such that} \\ \Delta_2, a : \omega, \Delta_3 \longrightarrow \Omega_1, a : \omega, \Omega_3 \\ \wedge \forall \hat{a} \in \text{unsolved}(\Delta_3), \text{ the solution for } \hat{a} \text{ in } \Omega_3 \text{ is } [\Omega_2]\hat{a} \\ [\Omega_1, a : \omega, \Omega_3](\Delta_2, a : \omega, \Delta_3) \Vdash^{\text{kc}} \sigma \Leftarrow \star \rightsquigarrow [\Omega_1, a : \omega, \Omega_3]\mu \\ [\Omega_1, a : \omega, \Omega_3](\Delta_2, a : \omega, \Delta_3) \end{array} $	<p>Lemma F.15 By Lemma F.29 As proved in Lemma F.15 Given by Lemma F.32 By Part 2 By Lemma F.29 By rule A-CTXE-TVAR-TT</p> <p>Part 2</p>
---	--

$$\begin{array}{l}
= [\Omega_1, a : \omega](\Delta_2, a : \omega) \\
= [\Omega_1]\Delta_2, a : [\Omega_1]\omega \\
= [\Omega_1]\Delta_2, a : [\Omega]\omega \\
= [\Omega_1]\Delta, a : [\Omega]\omega \\
= [\Omega]\Delta, a : [\Omega]\omega \\
[\Omega_1, a : \omega, \Omega_3]\mu \\
= [\Omega_1, \Omega_2](\Delta_3)\mu \\
= [\Omega](\Delta_3)\mu \\
[\Omega]\Delta, a : [\Omega]\omega \stackrel{\text{kc}}{\sigma} \Leftarrow \star \rightsquigarrow [\Omega](\Delta_3)\mu \\
[\Omega]\Delta \stackrel{\text{k}}{\forall} a : \kappa. \sigma : \star \rightsquigarrow \forall a : [\Omega]\omega. [\Omega](\Delta_3)\mu \\
[\Omega]\Delta \stackrel{\text{k}}{(\forall} a : \kappa. \sigma) : \star \rightsquigarrow [\Omega](\forall a : \omega. (\Delta_3)\mu)
\end{array}
\quad \left| \begin{array}{l}
\text{By Lemma F.44} \\
\text{By definition} \\
\text{By Lemma F.31} \\
\text{By Lemma F.45} \\
\text{By Lemma F.44} \\
\text{By the way } \Omega_3 \text{ is constructed} \\
\text{By equations} \\
\text{By rule KTT-FORALL} \\
\text{By substitution}
\end{array} \right.$$

- The case for rule **A-KTT-FORALLI** is similar to the previous case.

The notable thing is that we use the solution of $\widehat{\alpha}$ (as in the rule **A-KTT-FORALLI**) in Ω as the ω in rule **KTT-FORALL**.

Part 2 We have

$$\frac{\text{A-KC-SUB} \quad \Delta \stackrel{\text{k}}{\sigma} : \eta \rightsquigarrow \mu_1 + \Delta_1 \quad \Delta_1 \stackrel{\text{inst}}{\mu_1} : [\Delta_1]\eta \sqsubseteq [\Delta_1]\omega \rightsquigarrow \mu_2 + \Delta_2}{\Delta \stackrel{\text{kc}}{\sigma} \Leftarrow \omega \rightsquigarrow \mu_2 + \Delta_2}$$

Follows directly from Part 1 and soundness of instantiation (Lemma F.55).

Part 3 By induction on the judgment.

- Case

$$\frac{\text{A-KAPP-TT-ARROW} \quad \Delta \stackrel{\text{kc}}{\tau} \Leftarrow \omega_1 \rightsquigarrow \rho_2 + \Theta}{\Delta \stackrel{\text{kapp}}{(\rho_1 : \omega_1 \rightarrow \omega_2)} \bullet \tau : \omega_2 \rightsquigarrow \rho_1 \rho_2 + \Theta}$$

$$[\Omega]\Delta \stackrel{\text{inst}}{\rho_1} : [\Omega]\rho_1 : [\Omega]\omega_1 \rightarrow [\Omega]\omega_2 \sqsubseteq [\Omega]\omega_1 \rightarrow [\Omega]\omega_2 \rightsquigarrow [\Omega]\rho_1 \quad \left| \begin{array}{l} \text{By rule INST-REFL} \\ \text{Part 2} \end{array} \right.$$

$$[\Omega]\Delta \stackrel{\text{kc}}{\tau} \Leftarrow [\Omega]\omega_1 \rightsquigarrow [\Omega]\rho_2$$

- Case

$$\frac{\text{A-KAPP-TT-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \stackrel{\text{kapp}}{(\rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}])} \bullet \tau : \omega \rightsquigarrow \rho + \Theta}{\Delta \stackrel{\text{kapp}}{(\rho_1 : \forall a : \omega_1. \eta)} \bullet \tau : \omega \rightsquigarrow \rho + \Theta}$$

$$[\Omega](\Delta, \widehat{\alpha} : \omega_1) \stackrel{\text{inst}}{(\rho_1 @ \widehat{\alpha})} : ([\Omega](\eta[a \mapsto \widehat{\alpha}]) \sqsubseteq \omega' \rightarrow [\Omega]\omega \rightsquigarrow \rho_3) \quad \left| \begin{array}{l} \text{I.H.} \\ \text{Above} \end{array} \right.$$

$$[\Omega](\Delta, \widehat{\alpha} : \omega_1) \stackrel{\text{kc}}{\tau} \Leftarrow \omega' \rightsquigarrow \rho_4 \wedge [\Omega]\rho = \rho_3 \rho_4$$

$$\Delta \rightarrow \Theta$$

$$\Delta \rightarrow \Omega$$

$$\Delta, \widehat{\alpha} : \omega_1 \rightarrow \Theta$$

$$\Delta, \widehat{\alpha} : \omega_1 \rightarrow \Omega$$

$$[\Omega]\Delta = [\Omega](\Delta, \widehat{\alpha} : \omega_1)$$

$$[\Omega]\Delta \stackrel{\text{inst}}{(\rho_1 @ ([\Omega]\widehat{\alpha}))} : ([\Omega](\eta[a \mapsto [\Omega]\widehat{\alpha}]) \sqsubseteq \omega' \rightarrow [\Omega]\omega \rightsquigarrow \rho_3)$$

$$[\Omega]\Delta \stackrel{\text{kc}}{\tau} \Leftarrow \omega' \rightsquigarrow \rho_4$$

$$\Delta, \widehat{\alpha} : \omega_1 \stackrel{\text{ela}}{\widehat{\alpha}} : [\Delta, \widehat{\alpha} : \omega_1]\omega_1$$

$$[\Omega](\Delta, \widehat{\alpha} : \omega_1) \stackrel{\text{ela}}{[\Omega]\widehat{\alpha}} : [\Omega](\Delta, \widehat{\alpha} : \omega_1)\omega_1$$

$$[\Omega]\Delta \stackrel{\text{ela}}{[\Omega]\widehat{\alpha}} : [\Omega](\Delta, \widehat{\alpha} : \omega_1)\omega_1$$

$$[\Omega]\Delta \stackrel{\text{ela}}{[\Omega]\widehat{\alpha}} : [\Omega]\omega_1$$

$$[\Omega]\Delta \stackrel{\text{inst}}{[\Omega]\rho_1} : \forall a : [\Omega]\omega_1. [\Omega]\eta \sqsubseteq \omega' \rightarrow [\Omega]\omega \rightsquigarrow \rho_3$$

By Lemma F.15
By Lemma F.32
By Lemma F.15
By Lemma F.32
By Lemma F.45
By equations and substitutions
By equations
By rule A-ELA-KUVAR
By Lemma F.57
By equations
By Lemma F.30
By rule INST-FORALL

- The case for rule **A-KAPP-TT-FORALL-INFER** is similar as the previous case.
- Case

A-KAPP-TT-KUVAR

$$\frac{\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \Vdash^{\text{kc}} \tau \Leftarrow \widehat{\alpha}_1 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash^{\text{kapp}} (\rho_1 : \widehat{\alpha}) \bullet \tau : \widehat{\alpha}_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}$$

$\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \longrightarrow \Theta$	By Lemma F.15
$\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \longrightarrow \Omega$	By Lemma F.32
$[\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2] \widehat{\alpha}$	By definition
$[\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2] (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2)$	By Lemma F.22
$[\Omega] \widehat{\alpha} = [\Omega] (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2)$	By rule INST-REFL
$[\Omega] (\Delta_1, \widehat{\alpha} : \omega, \Delta_2) \Vdash^{\text{inst}} [\Omega] \rho_1 : [\Omega] \widehat{\alpha} \sqsubseteq [\Omega] \widehat{\alpha}_1 \rightarrow [\Omega] \widehat{\alpha}_2 \rightsquigarrow [\Omega] \rho_1$	Part 2
$[\Omega] (\Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2) \Vdash^{\text{kc}} \tau \Leftarrow [\Omega] \widehat{\alpha}_1 \rightsquigarrow [\Omega] \rho_2$	By Lemma F.15
$\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Theta$	By Lemma F.32
$\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \longrightarrow \Omega$	By Lemma F.45
$[\Omega] (\Delta_1, \widehat{\alpha} : \omega, \Delta_2) \Vdash^{\text{kc}} \tau \Leftarrow [\Omega] \widehat{\alpha}_1 \rightsquigarrow [\Omega] \rho_2$	

□

Lemma F.57 (Soundness of Elaborated Kinding). *If Δ ok, and $\Delta \Vdash^{\text{ela}} \mu : \eta$, and $\Delta \longrightarrow \Omega$, then $[\Omega] \Delta \Vdash^{\text{ela}} [\Omega] \mu : [\Omega] \eta$.*

PROOF. By a straightforward induction on the derivation.

□

Lemma F.58 (Soundness of Typing Signature). *If Δ ok, and $\Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$, then $[\Omega] \Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$.*

PROOF. We have

A-SIG-TT

$$\frac{\begin{array}{l} \text{]}\sigma\text{[} \quad \overline{a_i^i} = \text{fkv}(\sigma) \quad \Omega, \{\overline{\alpha_i : \star}, a_i : \overline{\alpha_i^i}\} \Vdash^{\text{k}} \sigma : \star \rightsquigarrow \eta \dashv \Delta \\ \phi_1^c = \text{scoped_sort}(\overline{a_i : [\Delta] \overline{\alpha_i^i}}) \quad \widehat{\phi}_2^c = \text{unsolved}(\Delta) \quad \Delta \Leftarrow \overline{a_i^i} \end{array}}{\Omega \Vdash^{\text{sig}} \text{data } T : \sigma \rightsquigarrow T : \forall \{\phi_2^c\}. ((\forall \{\phi_1^c\}. [\Delta] \eta) [\widehat{\phi}_2^c \mapsto \phi_2^c])}$$

From $\Delta \Leftarrow \overline{a_i^i}$ we know that all unsolved unification variables in Δ do not depend on $\overline{a_i^i}$.

Given $\widehat{\phi}_2^c = \text{unsolved}(\Delta)$, we further know that $\widehat{\phi}_2^c$ only contains unsolved unification variable that do not depend on $\overline{a_i^i}$.

So ϕ_2^c only contains type variables that do not depend on any unification variable or $\overline{a_i^i}$.

By weakening, we can add ϕ_2^c into the kinding judgment, so we get $\Omega, \phi_2^c, \{\overline{\alpha_i : \star}, a_i : \overline{\alpha_i^i}\} \Vdash^{\text{k}} \sigma : \star \rightsquigarrow \eta \dashv \Delta_1$, where Δ_1 is identical to Δ except for the presence of ϕ_2^c .

Now, we solve all unsolved unification variable in Δ_1 (i.e., the domain of $\widehat{\phi}_2^c$) to its corresponding type variable in ϕ_2^c . We get a complete context Ω_1 and $\Delta_1 \longrightarrow \Omega_1$.

By Lemma F.15, we have $\Omega, \phi_2^c, \{\overline{\alpha_i : \star}, a_i : \overline{\alpha_i^i}\} \longrightarrow \Delta_1$.

So by Lemma F.32 we have $\Omega, \phi_2^c, \{\overline{\alpha_i : \star}, a_i : \overline{\alpha_i^i}\} \longrightarrow \Omega_1$.

By soundness of kinding (Lemma F.56), we know that $[\Omega_1] (\Omega, \phi_2^c, \{\overline{\alpha_i : \star}, a_i : \overline{\alpha_i^i}\}) \Vdash^{\text{k}} \sigma : \star \rightsquigarrow [\Omega_1] \eta$.

$[\Omega_1] (\Omega, \phi_2^c, \{\overline{\alpha_i : \star}, a_i : \overline{\alpha_i^i}\}) = [\Omega] \Omega, \phi_2^c, \phi_3^c [\widehat{\phi}_2^c \mapsto \phi_2^c]$, where ϕ_3^c is a well-formed order of ϕ_1^c .

And $[\Omega_1]\eta = ([\Delta]\eta)[\widehat{\phi}^c \mapsto \phi_2^c]$, because Δ contains all the solved unification variable in Ω_1 except for $\widehat{\phi}^c$.

Namely, $[\Omega]\Omega, \phi_2^c, \phi_3^c[\widehat{\phi}^c \mapsto \phi_2^c] \Vdash \sigma : \star \rightsquigarrow ([\Delta]\eta)[\widehat{\phi}^c \mapsto \phi_2^c]$. By reordering the context while preserving well-formedness, we have $[\Omega]\Omega, \phi_2^c, \phi_1^c[\widehat{\phi}^c \mapsto \phi_2^c] \Vdash \sigma : \star \rightsquigarrow ([\Delta]\eta)[\widehat{\phi}^c \mapsto \phi_2^c]$.

By the kinding rule we can get $[\Omega]\Omega, \phi_2^c \Vdash \forall\{\phi_1^c[\widehat{\phi}^c \mapsto \phi_2^c]\}.\sigma : \star \rightsquigarrow \forall\{\phi_1^c[\widehat{\phi}^c \mapsto \phi_2^c]\}.\widehat{([\Delta]\eta)[\widehat{\phi}^c \mapsto \phi_2^c]}$. By substitution we get $[\Omega]\Omega, \phi_2^c \Vdash \forall\{\phi_1^c[\widehat{\phi}^c \mapsto \phi_2^c]\}.\sigma : \star \rightsquigarrow (\forall\{\phi_1^c\}.\widehat{([\Delta]\eta)[\widehat{\phi}^c \mapsto \phi_2^c]})$.

To prove the rule, our goal is to prove all preconditions in

$$\frac{\text{SIG-TT} \quad \text{[}\sigma\text{]} \quad \phi \in \mathcal{Q}(\sigma) \quad \phi_1^c \in \mathcal{Q}(\forall\phi^c.\eta) \quad \Sigma, \phi_1^c \Vdash \forall\phi.\sigma : \star \rightsquigarrow \forall\phi^c.\eta \quad |\phi| = |\phi^c|}{\Sigma \Vdash^{\text{sig}} \text{data } T : \sigma \rightsquigarrow T : \forall\{\phi_1^c\}.\forall\{\phi^c\}.\eta}$$

We have $\text{[}\sigma\text{]}$ as given. We claim that ϕ_1^c fits $\phi(\phi^c)$, and ϕ_2^c fits ϕ_1^c .

We first prove ϕ_1^c fits ϕ . Because $\phi_1^c \in \text{scoped_sort}(a_i : [\Delta]\widehat{a}_i^i)$, obviously ϕ_1^c is one of the well-formed permutation of \widehat{a}_i^i , namely the free kind binder of σ .

We then prove ϕ_2^c fits ϕ_1^c . That requires us to prove that ϕ_2^c is the free kind binder of $(\forall\{\phi_1^c\}.\widehat{([\Delta]\eta)[\widehat{\phi}^c \mapsto \phi_2^c]})$. Because $\widehat{\phi}_2^c = \text{unsolved}(\Delta)$, by Lemma F.52, we know every unsolved unification variable in $\widehat{\phi}_2^c$ either appears in $[\Delta]\eta$, or appears in ϕ_1^c . For sure $[\Delta]\eta$ and ϕ_1^c cannot contain more unsolved unification variable than $\widehat{\phi}_2^c$ or otherwise it would be ill-formed. Namely, $\widehat{\phi}_2^c$ are the free unification variables of $[\Delta]\eta$ and ϕ_1^c . By substituting $\widehat{\phi}_2^c$ with ϕ_2^c , we know that ϕ_2^c are the free kind binder in $(\forall\{\phi_1^c\}.\widehat{([\Delta]\eta)[\widehat{\phi}_2^c \mapsto \phi_2^c]})$.

By now we have proved all the preconditions and we conclude that $[\Omega]\Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta$. □

Lemma F.59 (Soundness of Typing Data Constructor Decl.). *If Δ ok, and $\Delta \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash_{([\Omega]\rho)}^{\text{dc}} \mathcal{D} \rightsquigarrow [\Omega]\mu$.*

PROOF. We have

$$\frac{\text{A-DC-TT} \quad \Delta, \blacktriangleright_D \Vdash^k \forall\phi.(\overline{\tau}_i^i \rightarrow \rho) : \star \rightsquigarrow \mu \dashv \Theta_1, \blacktriangleright_D, \Theta_2 \quad \widehat{\phi}^c = \text{unsolved}(\Theta_2)}{\Delta \Vdash_{\rho}^{\text{dc}} \forall\phi.D \overline{\tau}_i^i \rightsquigarrow \forall\{\phi^c\}.\widehat{([\Theta_2]\mu)[\widehat{\phi}^c \mapsto \phi^c]} \dashv \Theta_1}$$

To prove our goal, we claim that ϕ^c fits the ϕ^c in

$$\frac{\text{DC-TT} \quad \phi^c \in \mathcal{Q}(\mu \setminus_{\Sigma, \overline{\tau}_i^i}) \quad \Sigma, \phi^c \Vdash \forall\phi.\overline{\tau}_i^i \rightarrow \rho : \star \rightsquigarrow \mu}{\Sigma \Vdash_{\rho}^{\text{dc}} \forall\phi.D \overline{\tau}_i^i \rightsquigarrow \forall\{\phi^c\}.\mu}$$

We prove this by Lemma F.52 and the similar reasoning as in Lemma F.58.

The important thing to note is $\widehat{\phi}^c$ only contains unsolved unification variables in μ .

Note that μ might contain unsolved unification variables in Θ_1 . Then they must be the dependency of unsolved unification variables in Δ . And those are not unification variables that we should generalize over. □

Lemma F.60 (Soundness of Typing Datatype Decl.). *If Δ ok, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, and $\Theta \longrightarrow \Omega$, then $[\Omega]\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow [\Omega]\Gamma$.*

PROOF. We have

A-DT-TT

$$\frac{(T : \forall\{\phi_1^c\}. \forall\phi_2^c. \omega) \in \Delta \quad \Delta, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i \Vdash^\mu [\Delta]\omega \approx (\overline{\alpha_i^i} \rightarrow \star) \vdash \Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i = \omega_i}{\frac{\Theta_j, \phi_1^c, \phi_2^c, \overline{\alpha_i : \omega_i^i} \Vdash^{\text{dc}}_{(T \text{ @ } \phi_1^c \text{ @ } \phi_2^c \overline{\alpha_i^i})} \mathcal{D}_j \rightsquigarrow \mu_j \vdash \Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{\alpha_i : \omega_i^i}^j}{\Delta \Vdash^{\text{dt}} \mathbf{data} T \overline{\alpha_i^i} = \overline{\mathcal{D}_j}^{j \in 1..n} \rightsquigarrow D_j : \forall\{\phi_1^c\}. \forall\phi_2^c. \forall \overline{\alpha_i : \omega_i^i}^j . \mu_j \vdash \Theta_{n+1}}}$$

$\Delta, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i \longrightarrow \Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i = \omega_i$	By Lemma F.10
$\Delta \longrightarrow \Theta_1$	By Lemma F.29
$\frac{\Theta_j, \phi_1^c, \phi_2^c, \overline{\alpha_i : \omega_i^i} \longrightarrow \Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{\alpha_i : \omega_i^i}^j}{\Theta_j \longrightarrow \Theta_{j+1}^j}$	By Lemma F.18
$\Theta_{n+1} \longrightarrow \Omega$	By Lemma F.29
$\Theta_1 \longrightarrow \Omega$	Given
$\Delta \longrightarrow \Omega$	By Lemma F.32
$\Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i \longrightarrow \Omega, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i = \omega_i$	By definition
$[\Omega, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i]([\Delta]\omega) = [\Omega, \phi_1^c, \phi_2^c, \overline{\alpha_i : \star}^i] (\overline{\alpha_i^i} \rightarrow \star)$	By Lemma F.54
$[\Omega]([\Delta]\omega) = [\Omega] \overline{\omega_i^i} \rightarrow \star$	By definition and freshness
$[\Omega]\omega = [\Omega] \overline{\omega_i^i} \rightarrow \star$	By Lemma F.30
$(T : \forall\{\phi_1^c\}. \forall\phi_2^c. [\Omega] \overline{\omega_i^i} \rightarrow \star) \in [\Omega]\Delta$	By Lemma F.39
$\overline{\Theta_{j+1}} \longrightarrow \Omega^j$	By Lemma F.32
$\frac{\Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{\alpha_i : \omega_i^i} \longrightarrow \Omega, \phi_1^c, \phi_2^c, \overline{\alpha_i : \omega_i^i}^j}{[\Omega]\Delta, \phi_1^c, \phi_2^c, \overline{\alpha_i : [\Omega] \omega_i^i} \Vdash^{\text{dc}}_{(T \text{ @ } \phi_1^c \text{ @ } \phi_2^c \overline{\alpha_i^i})} \mathcal{D}_j \rightsquigarrow [\Omega]\mu_j}$	By definition
$[\Omega]\Delta \Vdash^{\text{dt}} \mathbf{data} T \overline{\alpha_i^i} = \overline{\mathcal{D}_j}^j \rightsquigarrow D_j : [\Omega](\forall\{\phi_1^c\}. \forall\phi_2^c. \forall \overline{\alpha_i : \omega_i^i}^j . \mu_j)$	By Lemma F.59, and Lemma F.45
	By rule DT-TT

□

Lemma F.61 (Soundness of Typing Program). *If $\Omega; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \mu$, then $[\Omega]\Omega; [\Omega]\Gamma \Vdash^{\text{pgm}} \text{pgm} : [\Omega]\mu$.*

PROOF. By induction on the derivation.

- Case

$$\frac{\text{A-PGM-EXPR} \quad [\Omega]\Omega; [\Omega]\Gamma \vdash e : \sigma}{\Omega; \Gamma \Vdash^{\text{pgm}} e : \sigma}$$

The goal holds directly.

- Case

$$\frac{\text{A-PGM-SIG} \quad \Omega \Vdash^{\text{sig}} \mathcal{S} \rightsquigarrow T : \eta \quad \Omega, T : \eta; \Gamma \Vdash^{\text{pgm}} \text{pgm} : \mu}{\Omega; \Gamma \Vdash^{\text{pgm}} \mathbf{sig} \mathcal{S}; \text{pgm} : \mu}$$

The goal holds directly from soundness of typing signature (Lemma F.58) and I.H..

• Case

A-PGM-DT-TT

$$\frac{\Theta_1 = \Omega, \overline{\widehat{\alpha}_i : \star}^i, \overline{T_i : \widehat{\alpha}_i}^i \quad \overline{\Theta_i \Vdash^{\text{dt}} \mathcal{T}_i \rightsquigarrow \Gamma_i \dashv \Theta_{i+1}}^i}{\overline{\widehat{\phi}_i^c = \text{unsolved}([\Theta_{n+1}]\widehat{\alpha}_i)}^i \quad \overline{\Theta_{n+1} \Vdash^{\text{gen}}_{\widehat{\phi}_i^c} ([\Theta_{n+1}](\Gamma_i[\widehat{\phi}_i^c \mapsto \phi_i^c]) \rightsquigarrow \Gamma'_i)}^i} \Omega, T_i : \forall \{\phi_i^c\}. ([\Theta_{n+1}]\widehat{\alpha}_i)[\widehat{\phi}_i^c \mapsto \phi_i^c] ; \Gamma, \Gamma'_i[\overline{T_i \mapsto T_i @ \phi_i^c}^i] \Vdash^{\text{pgm}} \text{pgm} : \mu \\ \Omega; \Gamma \Vdash^{\text{pgm}} \text{rec } \overline{\mathcal{T}_i}^{i \in 1..n}; \text{pgm} : \mu$$

The key is to prove that $\widehat{\phi}_i^c$ corresponds to the ϕ_i^c in rule **PGM-DT-TT**. The reasoning is similar to the one in Lemma **F.58**.

The key observation here is that, in typing datatype decl (rule **A-DT-TT**), the result context does not have new unification variables at the end. Therefore, all unsolved unification variable in Θ_{n+1} is in one of the free kind variable in $[\Theta_{n+1}]\widehat{\alpha}_i$. Once we have all the ϕ_i^c , the rest of preconditions follow straightforwardly. □

F.2.7 Principality.

Lemma F.62 (Completeness of Promotion). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\widehat{\alpha} \in \Delta$, and $\Delta \Vdash^{\text{ela}} \rho : \omega$, and $[\Delta]\widehat{\alpha} = \widehat{\alpha}$, and $[\Delta]\rho = \rho$, if ρ does not depend on $\widehat{\alpha}$ in the dependency graph, then there exists ρ_2 , Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$, and $\Delta \Vdash^{\text{pr}}_{\widehat{\alpha}} \rho \rightsquigarrow \rho_2 \dashv \Theta$.*

PROOF. By induction on the lexicographic order indicated in the proof of Theorem **F.50**.

The proof is essentially the same as Lemma **D.45**.

For case $\rho = \widehat{\beta}$, and the context $\Delta[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1]$, we have

$\Delta[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1] \Vdash^{\text{pr}}_{\widehat{\alpha}} [\Delta]\rho_1 \rightsquigarrow \rho_2 \dashv \Delta_1[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1]$	I.H.
$\wedge \Delta_1[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1] \longrightarrow \Omega_1 \wedge \Omega \longrightarrow \Omega_1$	Above
$\Delta[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1] \Vdash^{\text{pr}}_{\widehat{\alpha}} \widehat{\beta} \rightsquigarrow \widehat{\beta}_2 \dashv \Delta_1[\widehat{\beta}_2 : \rho_2, \widehat{\alpha} : \omega][\widehat{\beta} : \rho_1 = \widehat{\beta}_2]$	By rule A-PR-KUVARR-TT
$\Delta_1[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1] = \Delta_{11}, \widehat{\alpha} : \omega, \Delta_{12}, \widehat{\beta} : \rho_1, \Delta_{13}$	Let
$\Delta_{11} \Vdash^{\text{ela}} \rho_2 : \star$	By Lemma F.8
$\Omega_1 = \Omega_{11}, \widehat{\alpha} : \omega = \rho_3, \Omega_{12}, \widehat{\beta} : \rho_1 = \rho_4, \Omega_{13} \wedge \Delta_{11} \longrightarrow \Omega_{11}$	By Lemma F.29
$[\Delta_1]\rho_2 = [\Delta_1]([\Delta]\rho_1)$	By Lemma F.53
$= [\Delta_1]\rho_1$	By Lemma F.30
$[\Omega_1]\rho_2 = [\Omega_1]\rho_1$	By Lemma F.30
$\Omega_1 \Vdash^{\text{ela}} \rho_4 : [\Omega_1]\rho_1$	By inversion and weakening
$\Omega_1 \Vdash^{\text{ela}} [\Omega_1]\rho_4 : [\Omega_1]\rho_1$	By Lemma F.24
$\Omega_1 \Vdash^{\text{ela}} [\Omega_1]\rho_4 : [\Omega_1]\rho_2$	By equation
$\Omega_{11} \Vdash^{\text{ela}} [\Omega_1]\rho_4 : [\Omega_1]\rho_2$	By strengthening
$\Omega_{11} \Vdash^{\text{ela}} \rho_2 : \star$	By Lemma F.22
$\Omega_{11} \Vdash^{\text{ela}} [\Omega_1]\rho_4 : [\Omega_{11}]\rho_2$	By Lemma F.31
$\Delta_1[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1] \longrightarrow \Omega_1[\widehat{\alpha} : \omega][\widehat{\beta} : \rho_1 = \rho_4]$	Given
$\Delta_1[\widehat{\beta}_2 : \rho_2, \widehat{\alpha} : \omega][\widehat{\beta} : \rho_1] \longrightarrow \Omega_1[\widehat{\beta}_2 : \rho_2 = [\Omega_1]\rho_4, \widehat{\alpha} : \omega][\widehat{\beta} : \rho_1 = \rho_4]$	By Lemma F.36
$\Delta_1[\widehat{\beta}_2 : \rho_2, \widehat{\alpha} : \omega][\widehat{\beta} : \rho_1 = \widehat{\beta}_2] \longrightarrow \Omega_1[\widehat{\beta}_2 : \rho_2 = [\Omega_1]\rho_4, \widehat{\alpha} : \omega][\widehat{\beta} : \rho_1 = \rho_4]$	By Lemma F.37
$\Omega \longrightarrow \Omega_1[\widehat{\beta}_2 : \rho_2 = [\Omega_1]\rho_4, \widehat{\alpha} : \omega][\widehat{\beta} : \rho_1 = \rho_4]$	By Lemmas F.32 and F.34

□

Lemma F.63 (Completeness of Unification). *Given Δ ok, and $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{ela}} \rho_1 : \omega$ and $\Delta \Vdash^{\text{ela}} \rho_2 : \omega$, and $[\Delta]\rho_1 = \rho_1$ and $[\Delta]\rho_2 = \rho_2$, if $[\Omega]\rho_1 = [\Omega]\rho_2$, then there exists Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\mu} \rho_1 \approx \rho_2 \div \Theta$.*

PROOF. By induction on the lexicographic order indicated in the proof of Theorem F.51. Then case analysis on ρ_1 and ρ_2 .

The proof is essentially the same as Lemma D.46.

For case $\rho_1 = \widehat{\alpha}$, and ρ_2 does not depend on $\widehat{\alpha}$ in the dependency graph, we have

$\Delta \Vdash_{\widehat{\alpha}}^{\text{pr}} \rho_2 \rightsquigarrow \rho_3 \div \Theta_1 \wedge \Theta_1 \longrightarrow \Omega_1 \wedge \Omega \longrightarrow \Omega_1$	By Lemma F.62
$\Theta_1 = \Theta_{11}, \widehat{\alpha} : \omega_3, \Theta_{12} \wedge \Theta_{11} \Vdash^{\text{ela}} \rho_3 : [\Theta_{11}]\omega$	By Lemma F.8
$[\Omega_1]\rho_2 = [\Omega_1]\rho_3$	By Lemma F.53
$\Omega_1 = \Omega_{11}, \widehat{\alpha} : \omega_3 = \rho_4, \Omega_{12} \wedge \Theta_{11} \longrightarrow \Omega_{11}$	Lemma F.29
$[\Omega]\widehat{\alpha} = [\Omega]\rho_2$	Given
$[\Omega_1]\widehat{\alpha} = [\Omega_1]\rho_2$	By Lemma F.30
$[\Omega_1]\rho_4 = [\Omega_1]\rho_2$	By definition
$[\Omega_1]\rho_4 = [\Omega_1]\rho_3$	By equations
$\Delta \longrightarrow \Omega_1$	By Lemma F.8 and Lemma F.32
$\Delta \Vdash^{\text{ela}} \widehat{\alpha} : \omega$	Given
$\Omega_1 \Vdash^{\text{ela}} \widehat{\alpha} : [\Omega_1]\omega$	By Lemma F.22
$[\Omega_1]\omega_3 = [\Omega_1]\omega$	By inversion
$[\Omega_{11}]\omega_3 = [\Omega_{11}]\omega$	By Lemma F.31
$[\Omega_{11}](\Theta_{11}\omega_3) = [\Omega_{11}](\Theta_{11}\omega)$	By Lemma F.30
$\Theta_{11} \Vdash^{\mu} [\Theta_{11}]\omega_3 \approx [\Theta_{11}]\omega \div \Theta_2 \wedge \Theta_2 \longrightarrow \Omega_2 \wedge \Omega_{11} \longrightarrow \Omega_2$	I.H.
$\Delta \Vdash^{\mu} \widehat{\alpha} \approx \rho_2 \div \Theta_2, \widehat{\alpha} : \omega_3 = \rho_3, \Theta_{12}$	By rule A-U-KVARL-TT
$\Theta_2, \widehat{\alpha} : \omega_3 = \rho_3, \Theta_{12} \longrightarrow \Omega_2, \widehat{\alpha} : \omega_3 = \rho_4, \Omega_{12}$	By Lemma F.36, Lemma F.38, Lemma F.37
$\Omega \longrightarrow \Omega_2, \widehat{\alpha} : \omega_3 = \rho_4, \Omega_{12}$	Similarly

What if ρ_2 depends on $\widehat{\alpha}$? For that to be possible, according to the dependency graph, we have either $\widehat{\alpha} = \star$ or $\widehat{\alpha} = \rightarrow$. Then for the unification constrain to be solvable, we have either $\rho_2 = \star$, $\rho_2 = \rightarrow$, or $\rho_2 = \widehat{\beta}$. When $\rho_2 = \star$ or $\rho_2 = \rightarrow$, we know ρ_2 does not depend on $\widehat{\alpha}$ at all. When $\rho_2 = \widehat{\beta}$, because we know that the context is well-formed, if $\widehat{\beta}$ depends on $\widehat{\alpha}$, we must have $\widehat{\alpha}$ not depending on $\widehat{\beta}$. So we can solve the case using rule A-U-KVARR-TT.

The case when the variable in a local scope is similar. □

Lemma F.64 (Completeness of Instantiation). *Given $\Delta \longrightarrow \Omega$, and $\Delta \Vdash^{\text{ela}} \rho : \eta$ and $\Delta \Vdash^{\text{ela}} \omega : \star$, and $[\Delta]\eta = \eta$ and $[\Delta]\omega = \omega$, if $[\Omega]\Delta \Vdash^{\text{inst}} [\Omega]\rho_1 : [\Omega]\eta \sqsubseteq [\Omega]\omega \rightsquigarrow \rho_2$, then there exists ρ'_2 , Θ and Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$ and $\Delta \Vdash^{\text{inst}} \rho_1 : \eta \sqsubseteq \omega \rightsquigarrow \rho'_2 \div \Theta$, and $[\Omega']\rho'_2 = \rho_2$.*

PROOF. By induction on the declarative instantiation.

- Case

INST-REFL

$$\frac{}{\Sigma \Vdash^{\text{inst}} \mu : \omega \sqsubseteq \omega \rightsquigarrow \mu}$$

Follows directly from rule A-INST-REFL and Lemma F.63.

- Case

INST-FORALL

$$\frac{\Sigma \Vdash^{\text{ela}} \rho : \omega_1 \quad \Sigma \Vdash^{\text{inst}} \mu_1 @\rho : \eta[a \mapsto \rho] \sqsubseteq \omega_2 \rightsquigarrow \mu_2}{\Sigma \Vdash^{\text{inst}} \mu_1 : \forall a : \omega_1. \eta \sqsubseteq \omega_2 \rightsquigarrow \mu_2}$$

We case analyze η , and it can only be of the shape $\forall a : \omega_2. \eta_2$, and $[\Omega]\omega_2 = \omega_1$ and $[\Omega]\eta_2 = \eta$. From hypothesis we get $[\Omega]\Delta \stackrel{\text{inst}}{\vdash} ([\Omega]\mu_1 @ [\Omega]\rho) : [\Omega]\mu[a \mapsto [\Omega]\rho] \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu_2$.

By substitution, $[\Omega]\Delta \stackrel{\text{inst}}{\vdash} [\Omega](\mu_1 @ \rho) : [\Omega](\mu[a \mapsto \rho]) \sqsubseteq [\Omega]\omega_2 \rightsquigarrow [\Omega]\mu_2$.

By definition, $[\Omega, \widehat{\alpha} : \omega_1 = \rho]\Delta \stackrel{\text{inst}}{\vdash} [\Omega, \widehat{\alpha} : \omega_1 = \rho](\mu_1 @ \widehat{\alpha}) : [\Omega, \widehat{\alpha} : \omega_1 = \rho](\mu[a \mapsto \widehat{\alpha}]) \sqsubseteq [\Omega, \widehat{\alpha} : \omega_1 = \rho]\omega_2 \rightsquigarrow [\Omega, \widehat{\alpha} : \omega_1 = \rho]\mu_2$.

The goal follows directly from I.H., and rule **A-INST-FORALL**.

- The case for rule **INST-FORALL-INFER** is similar to the previous case. □

Lemma F.65 (Principality of Kinding).

- Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \stackrel{\text{k}}{\vdash} \sigma : \eta \rightsquigarrow \mu$, and $\Delta \stackrel{\text{k}}{\vdash} \sigma : \eta' \rightsquigarrow \mu' \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$. Moreover, $[\Omega']\eta' = \eta$. Furthermore, if μ and μ' are monotypes, then $[\Omega']\mu' = \mu$.
- Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \stackrel{\text{kc}}{\vdash} \sigma \Leftarrow [\Omega]\eta \rightsquigarrow \mu$, and $\Delta \stackrel{\text{kc}}{\vdash} \sigma \Leftarrow \eta \rightsquigarrow \mu' \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$. Furthermore, if μ and μ' are monotypes, then $[\Omega']\mu' = \mu$.
- Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \stackrel{\text{inst}}{\vdash} [\Omega]\rho_1 : [\Omega]\eta \sqsubseteq (\omega_1 \rightarrow \omega_2) \rightsquigarrow \rho_3$, and $[\Omega]\Delta \stackrel{\text{kc}}{\vdash} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_4$ and $\Delta \stackrel{\text{kapp}}{\vdash} (\rho_1 : \eta) \bullet \tau : \omega \rightsquigarrow \rho_2 \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$. Moreover, $[\Omega']\omega = \omega_2$. Further, $[\Omega']\rho_2 = \rho_3 \rho_4$.

PROOF. From this lemma, we make use of **Any** to ensure every algorithmic context can be extended to a complete context. The existence of **Any** does not affect at all how this lemma is used. By induction on the algorithmic kinding.

Part 1 • The case for rules **A-KTT-STAR**, **A-KTT-NAT**, **A-KTT-VAR**, **A-KTT-TCON**, and **A-KTT-ARROW** follows trivially by picking $\Theta = \Delta$, and $\Omega' = \Omega$.

- Case

A-KTT-FORALL

$$\frac{\Delta \stackrel{\text{kc}}{\vdash} \kappa \Leftarrow \star \rightsquigarrow \omega \dashv \Delta_1 \quad \Delta_1, a : \omega \stackrel{\text{kc}}{\vdash} \sigma \Leftarrow \star \rightsquigarrow \mu \dashv \Delta_2, a : \omega, \Delta_3 \quad \Delta_3 \hookrightarrow a}{\Delta \stackrel{\text{k}}{\vdash} \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega. [\Delta_3]\mu \dashv \Delta_2, \text{unsolved}(\Delta_3)}$$

$$[\Omega]\Delta \stackrel{\text{k}}{\vdash} \forall a : \kappa. \sigma : \star \rightsquigarrow \forall a : \omega_1. \mu_1$$

$$[\Omega]\Delta \stackrel{\text{kc}}{\vdash} \kappa \Leftarrow \star \rightsquigarrow \omega_1 \wedge [\Omega]\Delta, a : \omega_1 \stackrel{\text{kc}}{\vdash} \sigma \Leftarrow \star \rightsquigarrow \mu_1$$

$$\Delta_1 \longrightarrow \Omega_1 \wedge \Omega \longrightarrow \Omega_1 \wedge [\Omega_1]\omega = \omega_1$$

$$[\Omega_1, a : \omega](\Delta_1, a : \omega)$$

$$= [\Omega_1]\Delta_1, a : \omega_1$$

is a well-formed permutation of $[\Omega]\Delta$

$$[\Omega_1, a : \omega](\Delta_1, a : \omega) \stackrel{\text{kc}}{\vdash} \sigma \Leftarrow \star \rightsquigarrow \mu_1$$

$$\Delta_2, a : \omega, \Delta_3 \longrightarrow \Omega_2 \wedge \Omega_1, a : \omega \longrightarrow \Omega_2$$

$$\Delta_1, a : \omega \longrightarrow \Delta_2, a : \omega, \Delta_3$$

$$\Delta_1 \longrightarrow \Delta_2 \wedge \Delta_3 \text{ soft}$$

$$\Omega_2 = \Omega_{21}, a : \omega, \Omega_{22} \wedge \Delta_2 \longrightarrow \Omega_{21} \wedge \Omega_1 \longrightarrow \Omega_{21} \wedge \Omega_{22} \text{ soft}$$

construct Ω_{23} which contain same domain of $\text{unsolved}(\Delta_3)$

$$\Omega' = \Omega_{21}, \Omega_{23}$$

$$\Delta_2, \text{unsolved}(\Delta_3) \longrightarrow \Omega'$$

$$\Omega_1 \longrightarrow \Omega'$$

Given

By inversion

I.H.

By definition

By Lemma F.45 and Lemma F.46

Follows

I.H.

By Lemma F.15

By Lemma F.29

By Lemma F.29

Let

By rule **A-CTXE-SOLVE-TT**

By rule **A-CTXE-ADDSOLVED-TT**

- The case for rule **A-KTT-FORALLI** is similar as the previous case.

• Case

$$\frac{\text{A-KTT-APP} \quad \Delta \Vdash^k \tau_1 : \eta_1 \rightsquigarrow \rho_1 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\text{kapp}} (\rho_1 : [\Delta_1]\eta_1) \bullet \tau_2 : \omega \rightsquigarrow \rho \dashv \Theta}{\Delta \Vdash^k \tau_1 \tau_2 : \omega \rightsquigarrow \rho \dashv \Theta}$$

$[\Omega]\Delta \Vdash^k \tau_1 \tau_2 : \omega_2 \rightsquigarrow \rho_2 \rho_3$	Given
$[\Omega]\Delta \Vdash^k \tau_1 : \eta_2 \rightsquigarrow \rho_4 \wedge [\Omega]\Delta \Vdash^{\text{inst}} \rho_4 : \eta_2 \sqsubseteq \omega_1 \rightarrow \omega_2 \rightsquigarrow \rho_2$	By inversion
$\wedge [\Omega]\Delta \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega_1 \rightsquigarrow \rho_3$	I.H.
$\Delta_1 \rightarrow \Omega_1 \wedge \Omega \rightarrow \Omega_1 \wedge [\Omega_1]\rho_1 = \rho_4 \wedge [\Omega_1]\eta_1 = \eta_2$	By Lemma F.43 and Lemma F.46
$[\Omega_1]\Delta$ is a well-formed permutation of $[\Omega]\Delta$	By equations
$[\Omega_1]\Delta \Vdash^{\text{inst}} [\Omega_1]\rho_1 : [\Omega_1]\eta_1 \sqsubseteq \omega_1 \rightarrow \omega_2 \rightsquigarrow \rho_2$	By equations
$[\Omega_1]\Delta \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega_1 \rightsquigarrow \rho_3$	By Part 3
$\Theta \rightarrow \Omega' \wedge \Omega_1 \rightarrow \Omega' \wedge [\Omega']\rho = \rho_2 \rho_3 \wedge [\Omega']\omega = \omega_2$	By Lemma F.32
$\Omega \rightarrow \Omega'$	

• Case

$$\frac{\text{A-KTT-KAPP} \quad \Delta \Vdash^k \tau_1 : \eta \rightsquigarrow \rho_1 \dashv \Delta_1 \quad [\Delta_1]\eta = \forall a : \omega. \eta_2 \quad \Delta_1 \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega \rightsquigarrow \rho_2 \dashv \Delta_2}{\Delta \Vdash^k \tau_1 @ \tau_2 : \eta_2 [a \mapsto \rho_2] \rightsquigarrow \rho_1 @ \rho_2 \dashv \Delta_2}$$

$[\Omega]\Delta \Vdash^k \tau_1 @ \tau_2 : \mu_3 [a \mapsto \rho_3] \rightsquigarrow \rho_4 @ \rho_3$	Given
$[\Omega]\Delta \Vdash^k \tau_1 : \forall a : \omega_2. \mu_3 \rightsquigarrow \rho_4 \wedge [\Omega]\Delta \Vdash^{\text{kc}} \tau_2 \Leftarrow \omega_2 \rightsquigarrow \rho_3$	By inversion
$\Delta_1 \rightarrow \Omega_1 \wedge \Omega \rightarrow \Omega_1 \wedge [\Omega_1]\eta = \forall a : \omega_2. \mu_3 \wedge [\Omega_1]\rho_1 = \rho_4$	I.H.
$[\Omega_1]\Delta$ is a well-formed permutation of $[\Omega]\Delta$	By Lemma F.43 and Lemma F.46
$[\Omega_1]\eta = [\Omega_1]([\Delta_1]\eta)$	By Lemma F.30
$= \forall a : [\Omega_1]\omega. [\Omega_1]\eta_2$	By definition
$= [\Omega_1]\eta = \forall a : \omega_2. \mu_3$	Given
$[\Omega_1]\omega = \omega_2$	Follows
$[\Omega_1]\eta_2 = \mu_3$	Follows
$[\Omega_1]\Delta \Vdash^{\text{kc}} \tau_2 \Leftarrow [\Omega_1]\omega \rightsquigarrow \rho_3$	By equations
$\Delta_2 \rightarrow \Omega' \wedge \Omega_1 \rightarrow \Omega' \wedge [\Omega']\rho_2 = \rho_3$	By Part 2
$\Omega \rightarrow \Omega'$	By Lemma F.32
$[\Omega'](\eta_2 [a \mapsto \rho_2])$	
$= ([\Omega']\eta_2)[a \mapsto [\Omega']\rho_2]$	By substitution
$= (\mu_3)[a \mapsto \rho_3]$	By Lemma F.41

- The case for rule **A-KTT-KAPP-INFER** is similar as the previous case.

Part 2 We have

$$\frac{\text{A-KC-SUB} \quad \Delta \Vdash^k \sigma : \eta \rightsquigarrow \mu_1 \dashv \Delta_1 \quad \Delta_1 \Vdash^{\text{inst}} \mu_1 : [\Delta_1]\eta \sqsubseteq [\Delta_1]\omega \rightsquigarrow \mu_2 \dashv \Delta_2}{\Delta \Vdash^{\text{kc}} \sigma \Leftarrow \omega \rightsquigarrow \mu_2 \dashv \Delta_2}$$

$[\Omega]\Delta \Vdash^{\text{kc}} \sigma \Leftarrow [\Omega]\omega \rightsquigarrow \mu_4$	Given
$[\Omega]\Delta \Vdash^k \sigma : \eta_3 \rightsquigarrow \mu_3$	By inversion
$[\Omega]\Delta \Vdash^{\text{inst}} \mu_3 : \eta_3 \sqsubseteq [\Omega]\omega \rightsquigarrow \mu_4$	By inversion
$\Delta_1 \rightarrow \Omega_1 \wedge \Omega \rightarrow \Omega_1 \wedge [\Omega_1]\eta = \eta_3$	I.H.
If μ_1 and μ_3 are monotypes, then $[\Omega_1]\mu_1 = \mu_3$	I.H.
$[\Omega_1]\Delta$ is a well-formed permutation of $[\Omega]\Delta$	By Lemma F.43 and Lemma F.46

$[\Omega_1]\omega = [\Omega]\omega$	By Lemma F.41
$[\Omega_1]\Delta_1 \overset{\mu_3}{\text{inst}} [\Omega_1]\eta \sqsubseteq [\Omega_1]\omega \rightsquigarrow \mu_4$	Follows
If μ_1 and μ_3 are monotypes	
$[\Omega_1]\Delta_1 \overset{\mu_1}{\text{inst}} [\Omega_1]\mu_1 : [\Omega_1]\eta \sqsubseteq [\Omega_1]\omega \rightsquigarrow \mu_4$	Follows
$[\Omega_1]\Delta_1 \overset{\mu_1}{\text{inst}} [\Omega_1]\mu_1 : [\Omega_1](\Delta_1)\eta \sqsubseteq [\Omega_1](\Delta_1)\omega \rightsquigarrow \mu_4$	By Lemma F.30
$\Omega_1 \longrightarrow \Omega' \wedge \Delta_2 \longrightarrow \Omega' \wedge [\Omega']\mu_2 = \mu_4$	By Lemma F.64
If μ_1 and μ_3 are polytypes	
then only rule INST-REFL and rule A-INST-REFL can apply	
$\Delta_2 = \Delta_1$	Follows
$\Omega' = \Omega_1$	Let

Part 3 • Case

$$\frac{\text{A-KAPP-TT-ARROW} \quad \Delta \Vdash^{\text{kc}} \tau \Leftarrow \omega_1 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \omega_1 \rightarrow \omega_2) \bullet \tau : \omega_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}$$

$[\Omega]\Delta \overset{\mu_1}{\text{inst}} [\Omega]\rho_1 : [\Omega]\omega_1 \rightarrow [\Omega]\omega_2 \sqsubseteq [\Omega]\omega_1 \rightarrow [\Omega]\omega_2 \rightsquigarrow [\Omega]\rho_1$	Given
$[\Omega]\Delta \overset{\mu_2}{\text{kc}} \tau \Leftarrow [\Omega]\omega_1 \rightsquigarrow \rho_4$	Given
The goal follows directly from Part 2	

• Case

$$\frac{\text{A-KAPP-TT-FORALL} \quad \Delta, \widehat{\alpha} : \omega_1 \Vdash^{\text{kapp}} (\rho_1 @ \widehat{\alpha} : \eta[a \mapsto \widehat{\alpha}]) \bullet \tau : \omega \rightsquigarrow \rho \dashv \Theta}{\Delta \Vdash^{\text{kapp}} (\rho_1 : \forall a : \omega_1. \eta) \bullet \tau : \omega \rightsquigarrow \rho \dashv \Theta}$$

$[\Omega]\Delta \overset{\mu_1}{\text{inst}} [\Omega]\rho_1 : \forall a : [\Omega]\omega_1. [\Omega]\eta \sqsubseteq \omega_3 \rightarrow \omega_4 \rightsquigarrow \rho_3$	Given
$[\Omega]\Delta \overset{\mu_2}{\text{kc}} \tau \Leftarrow \omega_3 \rightsquigarrow \rho_4$	Given
$[\Omega]\Delta \overset{\mu_3}{\text{ela}} \rho_5 : [\Omega]\omega_1$	By inversion
$[\Omega]\Delta \overset{\mu_4}{\text{inst}} [\Omega]\rho_1 @ \rho_5 : ([\Omega]\eta)[a \mapsto \rho_5] \sqsubseteq \omega_3 \rightarrow \omega_4 \rightsquigarrow \rho_3$	By inversion
$[\Omega, \widehat{\alpha} : \omega_1 = \rho_5](\Delta, \widehat{\alpha} : \omega_1) \overset{\mu_5}{\text{inst}}$	
$[\Omega, \widehat{\alpha} : \omega_1 = \rho_5](\rho_1 @ \widehat{\alpha}) : [\Omega, \widehat{\alpha} : \omega_1 = \rho_5](\eta[a \mapsto \widehat{\alpha}]) \sqsubseteq \omega_3 \rightarrow \omega_4 \rightsquigarrow \rho_3$	By definition
The goal follows from I.H.	

• The case for rule **A-KAPP-TT-FORALL-INFER** is similar as the previous case.

• Case

$$\frac{\text{A-KAPP-TT-KUVAR} \quad \Delta_1, \widehat{\alpha}_1 : \star, \widehat{\alpha}_2 : \star, \widehat{\alpha} : \omega = (\widehat{\alpha}_1 \rightarrow \widehat{\alpha}_2), \Delta_2 \Vdash^{\text{kc}} \tau \Leftarrow \widehat{\alpha}_1 \rightsquigarrow \rho_2 \dashv \Theta}{\Delta_1, \widehat{\alpha} : \omega, \Delta_2 \Vdash^{\text{kapp}} (\rho_1 : \widehat{\alpha}) \bullet \tau : \widehat{\alpha}_2 \rightsquigarrow \rho_1 \rho_2 \dashv \Theta}$$

As $\widehat{\alpha}$ can only be instantiated with monotypes, obviously the declarative instantiation judgment must be rule **INST-REFL**. Then the goal follows directly from Part 2. □

Lemma F.66 (Principality of Typing Data Constructor Declaration). *Given $\Delta \longrightarrow \Omega$, if $[\Omega]\Delta \overset{\text{dc}}{\rho} \mathcal{D} \rightsquigarrow \mu_1$, and $\Delta \Vdash^{\text{dc}} \mathcal{D} \rightsquigarrow \mu_2 \dashv \Theta$, then there exists Ω' such that $\Theta \longrightarrow \Omega'$, and $\Omega \longrightarrow \Omega'$.*

PROOF. We have

$$\frac{\text{A-DC-TT} \quad \Delta, \blacktriangleright_D \Vdash^{\text{k}} \forall \phi. (\overline{\tau}_i^i \rightarrow \rho) : \star \rightsquigarrow \mu \dashv \Theta_1, \blacktriangleright_D, \Theta_2 \quad \widehat{\phi}^c = \text{unsolved}(\Theta_2)}{\Delta \Vdash^{\text{dc}} \forall \phi. D \overline{\tau}_i^i \rightsquigarrow \forall \{\phi^c\}. (([\Theta_2]\mu)[\widehat{\phi}^c \mapsto \phi^c]) \dashv \Theta_1}$$

$[\Omega]\Delta \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu_1$	Given
$[\Omega]\Delta, \phi^c \Vdash^k \forall \phi. \overline{\tau}_i^i \rightarrow \rho : \star \rightsquigarrow \mu_1$	By inversion
$\Delta, \blacktriangleright_D \Vdash^k \forall \phi. (\overline{\tau}_i^i \rightarrow \rho) : \star \rightsquigarrow \mu \dashv \Theta_1, \blacktriangleright_D, \Theta_2$	Given
$\Delta, \blacktriangleright_D, \phi^c \Vdash^k \forall \phi. (\overline{\tau}_i^i \rightarrow \rho) : \star \rightsquigarrow \mu \dashv \Theta_1, \blacktriangleright_D, \phi^c, \Theta_2$	By weakening
$\Delta \rightarrow \Omega$	Given
$\Delta, \blacktriangleright_D, \phi^c \rightarrow \Omega, \blacktriangleright_D, \phi^c$	By definition
$[\Omega, \blacktriangleright_D, \phi^c](\Delta, \blacktriangleright_D, \phi^c) \Vdash^k \forall \phi. \overline{\tau}_i^i \rightarrow \rho : \star \rightsquigarrow \mu_1$	By definition
$\Theta_1, \blacktriangleright_D, \phi^c, \Theta_2 \rightarrow \Omega_1 \wedge \Omega, \blacktriangleright_D, \phi^c \rightarrow \Omega_1$	By Lemma F.65
$\Omega_1 = \Omega', \blacktriangleright_D, \Omega_{12} \wedge \Theta_1 \rightarrow \Omega_1 \wedge \Theta_1 \rightarrow \Omega' \wedge \Omega \rightarrow \Omega'$	By Lemma F.29

□

Lemma F.67 (Principality of Typing Datatype Declaration). *Given $\Delta \rightarrow \Omega$, if $[\Omega]\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Psi$, and $\Delta \Vdash^{\text{dt}} \mathcal{T} \rightsquigarrow \Gamma \dashv \Theta$, then there exists Ω' such that $\Theta \rightarrow \Omega'$, and $\Omega \rightarrow \Omega'$.*

PROOF. We have

A-DT-TT

$$\frac{(T : \forall \{\phi_1^c\}. \forall \phi_2^c. \omega) \in \Delta \quad \Delta, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star \Vdash^{\text{dt}} [\Delta]\omega \approx (\overline{\alpha}_i^i \rightarrow \star) \dashv \Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \omega_i}{\frac{\Theta_j, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i \Vdash^{\text{dc}} (T @_{\phi_1^c} @_{\phi_2^c} \overline{a}_i^i) \mathcal{D}_j \rightsquigarrow \mu_j \dashv \Theta_{j+1}, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i}{\Delta \Vdash^{\text{dt}} \mathbf{data} T \overline{a}_i^i = \overline{\mathcal{D}_j}^{j \in 1..n} \rightsquigarrow \overline{\mathcal{D}_j} : \forall \{\phi_1^c\}. \forall \phi_2^c. \forall \overline{a}_i : \overline{\omega}_i^i. \mu_j \dashv \Theta_{n+1}}}$$

$[\Omega]\Delta \Vdash_{\rho}^{\text{dc}} \mathcal{D} \rightsquigarrow \mu_1$	Given
$(T : (\forall \{\phi_1^c\}. \forall \phi_2^c. [\Omega]\omega)) \in [\Omega]\Delta$	Inversion
$[\Omega]\omega = \overline{\omega}'_i^i \rightarrow \star$	Inversion
$[\Omega]\Delta, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}'_i^i \Vdash^{\text{dc}} (T @_{\phi_1^c} @_{\phi_2^c} \overline{a}_i^i) \mathcal{D}_j \rightsquigarrow \mu_j$	Inversion
$\Delta \rightarrow \Omega$	Given
$\Delta, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star \rightarrow \Omega, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \overline{\omega}'_i^i$	By definition
$[\Omega, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \overline{\omega}'_i^i]\omega = [\Omega, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \overline{\omega}'_i^i](\overline{\alpha}_i^i \rightarrow \star)$	By substitution
$\Omega, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \overline{\omega}'_i^i \rightarrow \Omega_1$	By Lemma F.63
$\Theta_1, \phi_1^c, \phi_2^c, \overline{\alpha}_i : \star = \omega_i \rightarrow \Omega_1$	By Lemma F.63
$\Omega_1 = \Omega_2, \Omega_3 \wedge \Omega \rightarrow \Omega_2 \wedge \Theta_1 \rightarrow \Omega_2$	By Lemma F.29
$\Theta_1, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i \rightarrow \Omega_2, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i$	By definition
$[\Omega_2, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i](\Theta_1, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i)$	
$= [\Omega_2]\Theta_1, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i$	By definition
$= [\Omega_2]\Theta_1, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}'_i^i$	By Lemma F.30
is a well-formed permutation of $[\Omega]\Delta, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}'_i^i$	By Lemma F.46
$[\Omega]\Delta, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}'_i^i \Vdash^{\text{dc}} (T @_{\phi_1^c} @_{\phi_2^c} \overline{a}_i^i) \mathcal{D}_j \rightsquigarrow \mu_j$	Given
$[\Omega_2, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i](\Theta_1, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i) \Vdash^{\text{dc}} (T @_{\phi_1^c} @_{\phi_2^c} \overline{a}_i^i) \mathcal{D}_j \rightsquigarrow \mu_j$	Follows
$(\Theta_2, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i) \rightarrow \Omega_3$	By Lemma F.66
$(\Omega_2, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i) \rightarrow \Omega_3$	By Lemma F.66

Repeating the process for each j , we can finally get $(\Theta_{n+1}, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i) \rightarrow \Omega''$, and $(\Omega_{n+1}, \phi_1^c, \phi_2^c, \overline{a}_i : \overline{\omega}_i^i) \rightarrow \Omega''$.

$$\begin{array}{l} \Omega'' = \Omega', \Omega_0 \wedge \Theta_{n+1} \longrightarrow \Omega' \wedge \Omega_{n+1} \longrightarrow \Omega' \\ \Omega \longrightarrow \Omega' \end{array} \left| \begin{array}{l} \text{By Lemma F.29} \\ \text{By Lemma F.32} \end{array} \right.$$

□

Theorem F.68 (Principality of Typing a Datatype Declaration Group). *If $\Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\eta}_i^i; \overline{\Gamma}_i^i$, then whenever $[\Omega]\Omega \Vdash^{\text{grp}} \text{rec } \overline{\mathcal{T}}_i^i \rightsquigarrow \overline{\eta}_i^i; \overline{\Psi}_i^i$ holds, we have $[\Omega]\Omega \vdash [\Omega]\eta_i \leq \eta_i'$.*

PROOF. Given
PGM-DT-TT

$$\frac{\frac{\overline{\Sigma, \phi_i^c \text{ela } \omega_i : \star}^i \quad \overline{\phi_i^c \in Q(\omega_i)}^i \quad \overline{\Sigma, \cup \phi_i^c, T_i : \omega_i^i \text{dt } \mathcal{T}_i \rightsquigarrow \Psi_i}^i}{\overline{\Sigma, \cup \phi_i^c, T_i : \omega_i^i \Vdash_{\phi_i^c}^{\text{gen}} \Psi_i \rightsquigarrow \Psi_i'}^i} \quad \overline{\Sigma, T_i : \forall \{\phi_i^c\}. \omega_i; \Psi, \Psi_i' [T_i \mapsto T_i @ \phi_i^c]^i}^i \text{pgm} : \sigma}{\overline{\Sigma; \Psi \text{pgm } \text{rec } \overline{\mathcal{T}}_i^i; \text{pgm} : \sigma}^i}$$

A-PGM-DT-TT

$$\frac{\overline{\Theta_1 = \Omega, \overline{\alpha}_i : \star, T_i : \overline{\alpha}_i^i}^i \quad \overline{\Theta_i \text{dt } \mathcal{T}_i \rightsquigarrow \Gamma_i \dashv \Theta_{i+1}}^i}{\overline{\widehat{\phi}_i^c = \text{unsolved}([\Theta_{n+1}]\overline{\alpha}_i)}^i \quad \overline{\Theta_{n+1} \Vdash_{\phi_i^c}^{\text{gen}} ([\Theta_{n+1}](\Gamma_i[\widehat{\phi}_i^c \mapsto \phi_i^c]) \rightsquigarrow \Gamma_i')}^i} \quad \overline{\Omega, T_i : \forall \{\phi_i^c\}. ([\Theta_{n+1}]\overline{\alpha}_i)[\widehat{\phi}_i^c \mapsto \phi_i^c]}^i; \Gamma, \Gamma_i' [T_i \mapsto T_i @ \phi_i^c]^i}^i \text{pgm} : \mu}{\overline{\Omega; \Gamma \text{pgm } \text{rec } \overline{\mathcal{T}}_i^{i \in 1..n}}; \text{pgm} : \mu}^i}$$

Our goal is to prove that $[\Omega]\Delta \vdash \forall \{\phi_i^c\}. ([\Theta_{n+1}]\overline{\alpha}_i)[\widehat{\phi}_i^c \mapsto \phi_i^c] \leq \forall \{\phi_i^c\}. \omega_i$.

Similar as the proof in Lemma F.67, we can weaken the context Θ_i by adding $\cup \overline{\phi}_i^c$. By weakening we can get Θ'_{n+1} , which is exactly the same as Θ_{n+1} , except for the addition of $\cup \overline{\phi}_i^c$.

Let Ω_1 be $\Omega, \cup \overline{\phi}_i^c, \overline{\alpha}_i : \star = \omega_i, T_i : \overline{\alpha}_i^i$.

According to the definition, our goal is equivalent to prove that for some Ω' , we have $\Theta'_{n+1} \longrightarrow \Omega'$, and $[\Omega']\overline{\alpha}_i = \omega_i$. According to Lemma F.67, we can prove there is indeed a Ω' , such that $\Omega_1 \longrightarrow \Omega'$ and $\Theta'_{n+1} \longrightarrow \Omega'$. Moreover $[\Omega']\overline{\alpha}_i = [\Omega_1]\overline{\alpha}_i = \omega_i$ by Lemma F.41, so we are done. □

REFERENCES

- Andreas Abel and Brigitte Pientka. 2011. Higher-order dynamic pattern unification for dependent types and records. In *International Conference on Typed Lambda Calculi and Applications*. Springer, 10–26.
- P. B. Andrews. 1971. Resolution in type Theory. *Journal of Symbolic Logic* 36 (1971), 414–432.
- Richard S. Bird and Lambert Meertens. 1998. Nested datatypes. In *LNCS 1422: Proceedings of Mathematics of Program Construction*, Johan Jeuring (Ed.). Springer-Verlag, Marstrand, Sweden, 52–67. <http://www.cs.ox.ac.uk/people/richard.bird/online/BirdMeertens98Nested.pdf>
- Joachim Breitner, Richard A Eisenberg, Simon Peyton Jones, and Stephanie Weirich. 2016. Safe zero-cost coercions for Haskell. *Journal of Functional Programming* 26 (2016).
- L. Cardelli. 1986. *A polymorphic lambda-calculus with Type:Type*. Technical Report 10. SRC.
- Manuel M. T. Chakravarty, Gabriele Keller, and Simon Peyton Jones. 2005. Associated type synonyms. In *Proceedings of the Tenth ACM SIGPLAN International Conference on Functional Programming (ICFP '05)*. ACM, New York, NY, USA, 241–253. <https://doi.org/10.1145/1086365.1086397>
- Jesper Cockx, Dominique Devriese, and Frank Piessens. 2016. Unifiers as equivalences: proof-relevant unification of dependently typed data. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming (ICFP 2016)*. ACM, New York, NY, USA, 270–283. <https://doi.org/10.1145/2951913.2951917>
- Claudio Sacerdoti Coen. 2004. *Mathematical knowledge management and interactive theorem proving*. Ph.D. Dissertation. University of Bologna, 2004. Technical Report UBLCS 2004-5.

- Marco Comini, Ferruccio Damiani, and Samuel Vrech. 2008. On polymorphic recursion, type systems, and abstract interpretation. In *International Static Analysis Symposium*. Springer, 144–158.
- Luis Damas and Robin Milner. 1982. Principal type-schemes for functional programs. In *Proceedings of the 9th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '82)*. ACM, New York, NY, USA, 207–212. <https://doi.org/10.1145/582153.582176>
- Ferruccio Damiani. 2003. Rank 2 intersection types for local definitions and conditional expressions. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 25, 4 (2003), 401–451.
- Joshua Dunfield and Neelakantan R. Krishnaswami. 2013. Complete and easy bidirectional typechecking for higher-rank polymorphism. In *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming (ICFP '13)*. ACM, New York, NY, USA, 429–442. <https://doi.org/10.1145/2500365.2500582>
- Richard A Eisenberg. 2016. *Dependent types in haskell: Theory and practice*. Ph.D. Dissertation. University of Pennsylvania.
- Richard A. Eisenberg, Dimitrios Vytiniotis, Simon Peyton Jones, and Stephanie Weirich. 2014. Closed type families with overlapping equations. In *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '14)*. ACM, New York, NY, USA, 671–683. <https://doi.org/10.1145/2535838.2535856>
- Richard A Eisenberg, Stephanie Weirich, and Hamidhasan G Ahmed. 2016. Visible type application. In *European Symposium on Programming*. Springer, 229–254.
- Ronald Garcia and Matteo Cimini. 2015. Principal type schemes for gradual programs. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '15)*. ACM, New York, NY, USA, 303–315. <https://doi.org/10.1145/2676726.2676992>
- Warren D Goldfarb. 1981. The undecidability of the second-order unification problem. *Theoretical Computer Science* 13, 2 (1981), 225–230.
- Roberta Gori and Giorgio Levi. 2002. An experiment in type inference and verification by abstract interpretation. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*. Springer, 225–239.
- Roberta Gori and Giorgio Levi. 2003. Properties of a type abstract interpreter. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*. Springer, 132–145.
- Adam Gundry and Conor McBride. 2013. A tutorial implementation of dynamic pattern unification. *Unpublished draft* (2013).
- Adam Gundry, Conor McBride, and James McKinna. 2010. Type inference in context. In *Proceedings of the third ACM SIGPLAN workshop on Mathematically structured functional programming*. ACM, 43–54.
- Adam Michael Gundry. 2013. *Type inference, Haskell and dependent types*. Ph.D. Dissertation. University of Strathclyde.
- Fritz Henglein. 1993. Type inference with polymorphic recursion. *ACM Trans. Program. Lang. Syst.* 15, 2 (April 1993), 253–289. <https://doi.org/10.1145/169701.169692>
- J. Roger Hindley. 1969. The principal type-scheme of an object in combinatory logic. *Trans. Amer. Math. Soc.* 146 (1969), 29–60.
- G. Huet. 1973. A unification algorithm for typed lambda calculus. *Theoretical Computer Science* 1, 1 (1973), 27–57.
- Trevor Jim. 1996. What are principal typings and what are they good for?. In *Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. ACM, 42–53.
- Mark P Jones. 1995. A system of constructor classes: overloading and implicit higher-order polymorphism. *Journal of functional programming* 5, 1 (1995), 1–35.
- Mark P. Jones. 1999. Typing Haskell in Haskell. In *Proceedings of the 1999 Haskell Workshop (Haskell '99)*, Erik Meijer (Ed.). Paris, France, pp. 9–22. University of Utrecht Technical Report UU-CS-1999-28.
- Csongor Kiss, Susan Eisenbach, Tony Field, and Simon Peyton Jones. 2019. Higher-order type-level programming in Haskell. In *Proceedings of the 24th ACM SIGPLAN International Conference on Functional Programming (ICFP 2019)*. ACM.
- Didier Le Botlan and Didier Rémy. 2003. MLF: Raising ML to the Power of System F (ICFP '03). 12.
- Daan Leijen. 2009. Flexible types: robust type inference for first-class polymorphism. In *Proceedings of the 36th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '09)*. ACM, New York, NY, USA, 66–77. <https://doi.org/10.1145/1480881.1480891>
- Dale Miller. 1991. Unification of simply typed lambda-terms as logic programming. (1991).
- Alan Mycroft. 1984. Polymorphic type schemes and recursive definitions. In *International Symposium on Programming*. Springer, 217–228.
- Martin Odersky and Konstantin Läucher. 1996. Putting type annotations to work. In *Proceedings of the 23rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '96)*. ACM, New York, NY, USA, 54–67. <https://doi.org/10.1145/237721.237729>
- Simon Peyton Jones. 2003. *Haskell 98 language and libraries: the revised report*. Cambridge University Press.
- Simon Peyton Jones, Dimitrios Vytiniotis, Stephanie Weirich, and Mark Shields. 2007. Practical type inference for arbitrary-rank types. *Journal of Functional Programming* 17, 1 (2007), 1–82.

- Simon Peyton Jones, Dimitrios Vytiniotis, Stephanie Weirich, and Geoffrey Washburn. 2006. Simple unification-based type inference for GADTs. In *Proceedings of the Eleventh ACM SIGPLAN International Conference on Functional Programming (ICFP '06)*. ACM, New York, NY, USA, 50–61. <https://doi.org/10.1145/1159803.1159811>
- François Pottier and Didier Rémy. 2005. The essence of ML type inference. *Advanced Topics in Types and Programming Languages* (2005).
- Jason Reed. 2009. Higher-order constraint simplification in dependent type theory. In *Proceedings of the Fourth International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice*. ACM, 49–56.
- Didier Rémy and Boris Yakobowski. 2008. From ML to MLF: Graphic type constraints with efficient type inference. In *Proceedings of the 13th ACM SIGPLAN International Conference on Functional Programming (ICFP '08)*. ACM, New York, NY, USA, 63–74. <https://doi.org/10.1145/1411204.1411216>
- Tom Schrijvers, Simon Peyton Jones, Martin Sulzmann, and Dimitrios Vytiniotis. 2009. Complete and decidable type inference for GADTs. In *Proceedings of the 14th ACM SIGPLAN International Conference on Functional Programming (ICFP '09)*. ACM, New York, NY, USA, 341–352. <https://doi.org/10.1145/1596550.1596599>
- Alejandro Serrano, Jurriaan Hage, Dimitrios Vytiniotis, and Simon Peyton Jones. 2018. Guarded impredicative polymorphism. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2018)*. ACM, New York, NY, USA, 783–796. <https://doi.org/10.1145/3192366.3192389>
- Vincent Simonet and François Pottier. 2007. A constraint-based approach to guarded algebraic data types. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 29, 1 (2007), 1.
- Dimitrios Vytiniotis, Simon Peyton Jones, Tom Schrijvers, and Martin Sulzmann. 2011. OutsideIn (X) Modular type inference with local assumptions. *Journal of functional programming* 21, 4-5 (2011), 333–412.
- Dimitrios Vytiniotis, Stephanie Weirich, and Simon Peyton Jones. 2008. FPH: First-class polymorphism for Haskell (ICFP '08). 12.
- Stephanie Weirich, Pritam Choudhury, Antoine Voizard, and Richard A. Eisenberg. 2019. A Role for dependent types in Haskell. *Proc. ACM Program. Lang.* 3, ICFP, Article 101 (July 2019), 29 pages. <https://doi.org/10.1145/3341705>
- Stephanie Weirich, Justin Hsu, and Richard A. Eisenberg. 2013. System FC with Explicit Kind Equality. In *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming (ICFP '13)*. ACM, New York, NY, USA, 275–286. <https://doi.org/10.1145/2500365.2500599>
- Stephanie Weirich, Antoine Voizard, Pedro Henrique Azevedo de Amorim, and Richard A Eisenberg. 2017. A specification for dependent types in Haskell. In *Proceedings of the 22th ACM SIGPLAN International Conference on Functional Programming (ICFP '17)*. ACM.
- Hongwei Xi, Chiyan Chen, and Gang Chen. 2003. Guarded recursive datatype constructors. In *Proceedings of the 30th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '03)*. ACM, New York, NY, USA, 224–235. <https://doi.org/10.1145/604131.604150>
- Ningning Xie and Richard A Eisenberg. 2018. Coercion Quantification. In *Haskell Implementors's Workshop*.
- Brent A. Yorgey, Stephanie Weirich, Julien Cretin, Simon Peyton Jones, Dimitrios Vytiniotis, and José Pedro Magalhães. 2012. Giving Haskell a Promotion. In *Proceedings of the 8th ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI '12)*. ACM, New York, NY, USA, 53–66. <https://doi.org/10.1145/2103786.2103795>
- Beta Ziliani and Matthieu Sozeau. 2015. A Unification Algorithm for Coq Featuring Universe Polymorphism and Overloading. In *Proceedings of the 20th ACM SIGPLAN International Conference on Functional Programming (ICFP 2015)*. ACM, New York, NY, USA, 179–191. <https://doi.org/10.1145/2784731.2784751>