Iris: Higher-Order Concurrent Separation Logic

Lecture 10: Ghost State

Lars Birkedal

Aarhus University, Denmark

November 24, 2017

Overview

Earlier:

- Operational Semantics of $\lambda_{\mathrm{ref,conc}}$
  - e, $(h, e) \rightsquigarrow (h, e')$, and $(h, \mathcal{E}) \rightarrow (h', \mathcal{E}')$
- Basic Logic of Resources
  - $l \hookrightarrow v$, $P * Q$, $P \mathrel{-\!\!*} Q$, $\Gamma \mid P \vdash Q$
- Basic Separation Logic
  - $\{P\}\, e\, \{v.Q\}$: Prop, isList l xs, ADTs, foldr
- Later ($\triangleright$) and Persistent ($\square$) Modalities.
- Concurrency Intro and Invariants.

Today:

- Ghost State.
- Key Points:
  - Ghost (aka auxiliary / logical / abstract) state to capture more expressive invariants
  - Protocols as Invariants

Simple Motivating Example

- Last time we proved

$$\{\ell \hookrightarrow n\}\,(e \,||\, e); !\,\ell\,\{v.v \geq n\}$$

  where e is the program $\ell \leftarrow !\,\ell + 1$.
- Here is a more precise specification:

$$\{\ell \hookrightarrow n\}\,(e \,||\, e); !\,\ell\,\{v.v \geq n + 1\}$$

  which we ought to be able to show. But how ?
- So far, our invariants have limited expressive power.
- We could try to use an invariant such as

$$\ell \hookrightarrow n \vee \ell \hookrightarrow (n + 1)$$

- However, with what we have seen so far, we cannot ensure that once $\ell \hookrightarrow (n + 1)$ holds, the location ℓ will never point to n again.
- We will use *ghost state* to express this *protocol for how the shared state may evolve.*

Plan

1. We consider how to prove the above concrete example specification

$$\{\ell \hookrightarrow n\}\,(e \mid\mid e);!\,\ell\,\{v.v \geq n+1\}$$

   using (ad-hoc) ghost state.
2. Next we turn to general description of ghost state.

Example Ghost Resources

- We extend Iris with a new type GhostName of *ghost names*, typically denoted by γ.
- May think of ghost names as analogous to concrete heap locations but for abstract state of the program (no runtime significance).
- So ghost names are sometimes referred to as ghost variables.
- No operations on ghost names, they are introduced by allocation (analogous to references).
- To show example spec, we will use additional resources indexed by ghostnames: $\boxed{\mathrm{S}}^{\gamma}$ and $\boxed{\overline{\mathrm{F}}}^{\gamma}$.

Example Ghost Resources

The ghost resources satisfy:

F-DUPLICABLE

$$\overline{\boxed{\mathrm{F}}^{\gamma} \vdash \boxed{\mathrm{F}}^{\gamma} * \boxed{\mathrm{F}}^{\gamma}}$$

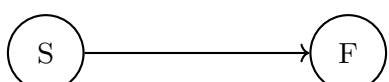
S-S-INCOMPATIBLE

$$\overline{\boxed{\mathrm{S}}^{\gamma} * \boxed{\mathrm{S}}^{\gamma} \vdash \mathsf{False}}$$

S-F-INCOMPATIBLE

$$\overline{\boxed{\mathrm{S}}^{\gamma} * \boxed{\mathrm{F}}^{\gamma} \vdash \mathsf{False}}$$

The invariant will start out in the "start state" $\boxed{\mathrm{S}}^{\gamma}$ and once the invariant is in the "finished state" $\boxed{\mathrm{F}}^{\gamma}$, it can never go back to the start state.
Conceptually, the tokens are used to encode the following transition system.



Example Ghost State: Hoare triples

Additionally, we need rules relating these tokens to Hoare triples:

$$\textsc{Ht-token-alloc}\quad \frac{T \in \{\mathrm{S}, \mathrm{F}\} \qquad S \vdash \left\{\exists \gamma.\, \boxed{T}^{\gamma} * P\right\} e \left\{v.Q\right\}}{S \vdash \{P\}\, e\, \{v.Q\}}$$

$$\textsc{Ht-token-update-pre}\quad \frac{S \vdash \left\{\boxed{\mathrm{F}}^{\gamma} * P\right\} e \left\{v.Q\right\}}{S \vdash \left\{\boxed{\mathrm{S}}^{\gamma} * P\right\} e \left\{v.Q\right\}}$$

$$\textsc{Ht-token-update-post}\quad \frac{S \vdash \{P\}\, e \left\{v.\boxed{\mathrm{S}}^{\gamma} * Q\right\}}{S \vdash \{P\}\, e \left\{v.\boxed{\mathrm{F}}^{\gamma} * Q\right\}}$$

Example Proof Idea

- We use the following invarint (parametrised by $\gamma \in \text{GhostName}$):

$$I(\gamma) = \exists m.\, \ell \hookrightarrow m * \left(\left(\boxed{\mathrm{S}}^{\gamma} \wedge m \geq n\right) \vee \left(\boxed{\mathrm{F}}^{\gamma} \wedge m \geq (n+1)\right)\right)$$

- Invariant will be allocated in start state, when ℓ is at least n.
- When a thread increases value stored at ℓ, invariant is updated to finished state.

Example Proof Idea

- We use the following invarint (parametrised by $\gamma \in \text{GhostName}$):

$$I(\gamma) = \exists m.\, \ell \hookrightarrow m * \left(\left(\boxed{\text{S}}^{\gamma} \wedge m \geq n \right) \vee \left(\boxed{\text{F}}^{\gamma} \wedge m \geq (n+1) \right) \right)$$

- Invariant will be allocated in start state, when ℓ is at least n.
- When a thread increases value stored at ℓ, invariant is updated to finished state.
- So let us begin proving the desired specification, which (recall) is:

$$\{\ell \hookrightarrow n\}\, (e \,||\, e);\, !\, \ell\, \{v.v \geq n+1\}$$

Example Proof

- We start by using rule HT-TOKEN-ALLOC and HT-EXIST, so SFTS

$$\left\{\boxed{\mathrm{S}}^{\gamma} * \ell \hookrightarrow n\right\}(e \mid\mid e); !\,\ell\,\{v.v \geq n+1\}.$$

- By sequencing rule HT-SEQ, SFTS

$$\left\{\boxed{\mathrm{S}}^{\gamma} * \ell \hookrightarrow n\right\} e \mid\mid e\left\{v.\boxed{\mathrm{F}}^{\gamma}\right\} \tag{1}$$

$$\left\{\boxed{\mathrm{F}}^{\gamma}\right\}\,!\,\ell\,\{v.v \geq n\}. \tag{2}$$

- Begin with the first triple.
- Use the invariant allocation rule HT-INV-ALLOC. Ok by rule of consequence HT-CSQ since $\boxed{\mathrm{S}}^{\gamma} * \ell \hookrightarrow n$ implies $I(\gamma)$. Hence SFTS:

$$\boxed{I(\gamma)}^{\iota} \vdash \{\mathsf{True}\}\, e \mid\mid e\left\{v.\boxed{\mathrm{F}}^{\gamma}\right\}.$$

Example Proof

- Since $\boxed{\mathrm{F}}^{\gamma} * \boxed{\mathrm{F}}^{\gamma}$ implies $\boxed{\mathrm{F}}^{\gamma}$ it suffices (by consequence) to use HT-PAR to show

$$\boxed{I(\gamma)}^{\iota} \vdash \{\mathsf{True}\}\, e \left\{v. \boxed{\mathrm{F}}^{\gamma}\right\}.$$

- Again, using the bind rule, we first need to prove

$$\boxed{I(\gamma)}^{\iota} \vdash \{\mathsf{True}\}\, !\,\ell \,\{v. v \geq n\}.$$

  Exercise!

- For the other premise of bind rule, TS

$$\boxed{I(\gamma)}^{\iota} \vdash \{m \geq n\}\, \ell \leftarrow (m+1) \left\{\_. \boxed{\mathrm{F}}^{\gamma}\right\}.$$

Example Proof

- To open the invariant we need an atomic expression.
- We use HT-BIND-DET to evaluate $m+1$ to a value using HT-OP. Then open invariant, and after using disjunction rule HT-DISJ and other structural rules SFTS:

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\triangleright(\ell \hookrightarrow m * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n)\right\} \ell \leftarrow (m+1) \left\{\_. \triangleright I(\gamma) * \boxed{\mathrm{F}}^{\gamma}\right\}$$
$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\triangleright(\ell \hookrightarrow m * \boxed{\mathrm{F}}^{\gamma} \wedge m \geq (n+1))\right\} \ell \leftarrow (m+1) \left\{\_. \triangleright I(\gamma) * \boxed{\mathrm{F}}^{\gamma}\right\}$$

- We only show the first, and leave the second one as an exercise. Note however that duplicability of $\boxed{\mathrm{F}}^{\gamma}$ is essential.

Examle Proof

- Using HT-FRAME-ATOMIC and HT-STORE we get

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\triangleright(\ell \hookrightarrow m * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n)\right\} \ell \leftarrow (m+1) \left\{v.v = () \wedge \ell \hookrightarrow (m+1) * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n\right\}$$

- Following up with HT-TOKEN-UPDATE-POST we get

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\triangleright(\ell \hookrightarrow m * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n)\right\} \ell \leftarrow (m+1) \left\{v.v = () \wedge \ell \hookrightarrow (m+1) * \boxed{\mathrm{F}}^{\gamma} \wedge m \geq n\right\}$$

- From this we easily get the required triple

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\triangleright(\ell \hookrightarrow m * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n)\right\} \ell \leftarrow (m+1) \left\{\_.\triangleright I(\gamma) * \boxed{\mathrm{F}}^{\gamma}\right\}$$

  using F-DUPLICABLE to create another copy of $\boxed{\mathrm{F}}^{\gamma}$ One of the copies is used to reestablish the invariant $I(\gamma)$, and the other remains in the postcondition.

Example Proof

- To conclude the proof, it remains to show

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\boxed{\mathrm{F}}^{\gamma}\right\}\, !\, \ell\, \{v.v \geq n+1\}$$

- Using HT-INV-OPEN together with structural rules, SFTS

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\boxed{\mathrm{F}}^{\gamma} * \triangleright(\ell \hookrightarrow m * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n)\right\}\, !\, \ell\, \{v.v \geq (n+1) \wedge \triangleright I(\gamma)\}$$
$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\boxed{\mathrm{F}}^{\gamma} * \triangleright(\ell \hookrightarrow m * \boxed{\mathrm{F}}^{\gamma} \wedge m \geq (n+1))\right\}\, !\, \ell\, \{v.v \geq (n+1) \wedge \triangleright I(\gamma)\}$$

- We again prove the first one and leave the second one as an exercise.
- Using HT-FRAME-ATOMIC and HT-LOAD we get

$$\boxed{I(\gamma)}^{\iota} \vdash \left\{\boxed{\mathrm{F}}^{\gamma} * \triangleright(\ell \hookrightarrow m * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n)\right\}\, !\, \ell\, \left\{v.v = m \wedge \ell \hookrightarrow m * \boxed{\mathrm{F}}^{\gamma} * \boxed{\mathrm{S}}^{\gamma} \wedge m \geq n\right\}$$

- Now by S-F-INCOMPATIBLE, the precondition is equivalent to $\triangleright$ False, that is, this case is *impossible*, and hence by HT-LATER-FALSE we get the desired triple.

Ghost State (More Generally)

- Need a general mechanism for defining and reasoning with ghost state.
- Remark: the set of heaps form a partial commutative monoid (Heap, $\cdot$, []).
- Idea now: we use certain kinds of *monoid-like structures* as abstract notions of resources.
- Empirically, many proof patterns can be described using these structures.

Commutative Semigroup and Monoid

- A *commutative semigroup* is a set $\mathcal{M}$ together with associative and commutative operation $(\cdot) : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$.
- A commutative semigroup is called a *commutative monoid* if there is a unit ε for $(\cdot)$: (*i.e.*, for all $m \in \mathcal{M}$, the property $m \cdot \varepsilon = \varepsilon \cdot m = m$ holds).
- The set $\mathcal{M}$ is called the *carrier*.
- Every semigroup yields a preorder by defining the *extension order* $a \preccurlyeq b$ as

$$a \preccurlyeq b \iff \exists c, b = a \cdot c.$$

  In words, $a \preccurlyeq b$ if *a is a part of b*.

Resource Algebra

- A *resource algebra* is a commutative semigroup $\mathcal{M}$ together with a subset $\mathcal{V} \subseteq \mathcal{M}$ of elements called *valid*, and a *partial* function $|\cdot| : \mathcal{M} \to \mathcal{M}$, called the *core*.
- The set of valid elements must satisfy

$$a \cdot b \in \mathcal{V} \Rightarrow a \in \mathcal{V}.$$

- The core is required to have the following properties.

$$|a| \text{ defined} \Rightarrow |a| \cdot a = a$$
$$|a| \text{ defined} \Rightarrow ||a|| = |a|$$
$$a \preccurlyeq b \land |a| \text{ defined} \Rightarrow |b| \text{ defined} \land |a| \preccurlyeq |b|.$$

- A resource algebra is *unital* if $\mathcal{M}$ is a commutative monoid with unit ε and the following properties hold.

$$\varepsilon \in \mathcal{V} \qquad\qquad |\varepsilon| = \varepsilon.$$

  In particular $|\varepsilon|$ is defined.

Example: Unital Resource Algebra of Heaps

- Carrier: $\mathit{Heap} \uplus \{\bot\}$.
- Composition of heaps is disjoint union, and if the heaps are not disjoint, then their composition is defined to be $\bot$.
- Composing $\bot$ with any other element yields $\bot$.
- The core is the constant function, mapping every element to the empty heap, which is the unit of the resource algebra.
- Every heap is valid, the only non-valid element being $\bot$.

More Examples of Resource Algebras

- Next 6 slides contain more examples of resource algebras / constructions of resource algebras.
- Very important for program proving – so study them carefully!
- However, to keep the overview, we skip them for now and move on to discuss how we use resource algebras in Iris.

Example: Discrete Resource Algebra

- Given set X, the carrier is $X \cup \{\bot\}$, for some element $\bot$ not in X.
- The operation $(\cdot)$ is defined by:
  - The non-trivial compositions are only

$$m \cdot m = m$$

    and otherwise $m \cdot n = \bot$.
- The core is the identity function.
- Every element apart from $\bot$ is valid.

Example: Products of Resource Algebras

- The product of resource algebras $(\mathcal{M}_1, |\cdot|_1, \mathcal{V}_1)$ and $(\mathcal{M}_2, |\cdot|_2, \mathcal{V}_2$ is defined componentwise:
- Carrier: the product of the carriers $\mathcal{M}_1 \times \mathcal{M}_2$.
- Operation:

$$(a, b) \cdot (a', b') = (a \cdot a', b \cdot b')$$

- Valid elements:

$$\mathcal{V}_\times = \{(a, b) \mid a \in \mathcal{V}_1, b \in \mathcal{V}_2\}.$$

- Core:

$$|(a, b)|_\times = \begin{cases} (|a|_1, |b|_2) & \text{if } |a|_1 \text{ and } |b|_2 \text{ defined} \\ \text{undefined} & \text{otherwise} \end{cases}$$

Example: Finite Map Resource Algebra

- Let $(\mathcal{M}, \mathcal{V}, |\cdot|)$ be a resource algebra.
- Define new resource algebra $\mathbb{N} \xrightarrow{\text{fin}} \mathcal{M}$, whose carrier is the set of partial functions $\mathbb{N} \to \mathcal{M}$ with finite domain and operation defined by:

$$(f \cdot g)(n) = \begin{cases} f(n) \cdot g(n) & \text{if } f(n) \text{ and } g(n) \text{ defined} \\ f(n) & \text{if } f(n) \text{ defined and } g(n) \text{ undefined} \\ g(n) & \text{if } g(n) \text{ defined and } f(n) \text{ undefined} \\ \text{undefined} & \text{otherwise} \end{cases}$$

- The set of valid finite maps is

$$\mathcal{V}_{\mathbb{N} \xrightarrow{\text{fin}} \mathcal{M}} = \{f \mid \forall n, f(n) \text{ defined} \Rightarrow f(n) \in \mathcal{V}\}$$

- The core is defined as

$$(|f|_{\mathbb{N} \xrightarrow{\text{fin}} \mathcal{M}})(n) = \begin{cases} |f(n)| & \text{if } f(n) \text{ and } |f(n)| \text{ defined} \\ \text{undefined} & \text{otherwise} \end{cases}$$

- Note that $\mathbb{N} \xrightarrow{\text{fin}} \mathcal{M}$ is always a *unital* resource algebra (the unit being the always undefined finite partial function).

Example: Exclusive Resource Algebra

- Given a set X, the exclusive resource algebra $\mathrm{Ex}(X)$ has carrier the set X with an additional element $\bot$.
- Operation: $x \cdot y = \bot$ for all x and y.
- The core is the always undefined function.
- The valid elements are elements of X.

Useful building block:

- The valid elements of $\mathbb{N} \xrightarrow{\text{fin}} \mathrm{Ex}\,(\mathit{Val})$ are precisely the heaps, and composition of these is exactly the same as the composition of heaps, if it is a valid element.
- Proof: exercise!

Example: Option Resource Algebra

- Given resource algebra $\mathcal{M}$, define the unital resource algebra $\mathcal{M}_?$.
- Carrier is the set $\mathcal{M}$ together with a new element ?.
- Operation: on elements of $\mathcal{M}$ it is inherited, and we additionally set $? \cdot x = x \cdot ? = x$, *i.e.,* ? is the unit.
- The set of valid elements is that of $\mathcal{M}$ and ?.
- Finally, the core operation is defined as

$$|?|_{\mathcal{M}_?} = ?$$

$$|x|_{\mathcal{M}_?} = \begin{cases} |x| & \text{if } |x| \text{ defined} \\ ? & \text{otherwise} \end{cases}$$

Example: Resource Algebra of Fractions

- Resource algebra fractions $\mathbb{Q}_{01}$.
- Carrier: the set of (strictly) positive rational numbers.
- Operation: addition
- Valid elements: only those q less than 1, *i.e.*, $\mathcal{V} = \{q \mid 0 < q \leq 1\}$.
- The core is always undefined.

Resource Algebras in Iris

- Extend logic with family of chosen resource algebras $\mathcal{M}_i$.
- We add the resource algebras, and its elements, the core function, and the property of the element being valid, as new types and new terms of the logic, together with all the equations for the operations.

$$\frac{\Gamma \vdash a : \mathcal{M}_i \qquad |a|_i \text{ defined}}{\Gamma \vdash |a|_i : \mathcal{M}_i} \qquad \frac{\Gamma \vdash a : \mathcal{M}_i}{\Gamma \vdash a \in \mathcal{V}_i : \mathsf{Prop}} \qquad \frac{\gamma \in \mathsf{GhostName} \qquad \Gamma \vdash a : \mathcal{M}_i}{\Gamma \vdash \boxed{a : \mathcal{M}_i}^{\gamma} : \mathsf{Prop}}$$

- The last rule introduces *ghost ownership assertion* $\boxed{a : \mathcal{M}_i}^{\gamma}$.
- We write it as $\boxed{a}^{\gamma}$ when $\mathcal{M}_i$ is clear from the context.

Ghost Ownership Rules

The rules of the ghost ownership assertion are as follows.

OWN-OP

$$\boxed{a : \mathcal{M}_i}^{\gamma} * \boxed{b : \mathcal{M}_i}^{\gamma} \dashv\vdash \boxed{a \cdot b : \mathcal{M}_i}^{\gamma}$$

OWN-VALID

$$\boxed{a : \mathcal{M}_i}^{\gamma} \vdash a \in \mathcal{V}_i$$

Finally, ghost ownership is persistent:

ALWAYS-CORE

$$\frac{\Gamma \vdash a : \mathcal{M}_i \qquad |a|_i \text{ defined}}{\boxed{a : \mathcal{M}_i}^{\gamma} \vdash \Box \boxed{|a|_i : \mathcal{M}_i}^{\gamma}}$$

Ghost Updates

- Iris always maintains the invariant that the ghost state obtained by composing the contributions of all threads is well-defined and valid.
- We call state changes that maintain this invariant *frame-preserving updates*.
- Definition: Given resource algebra $\mathcal{M}$, with valid elements $\mathcal{V}$, we define a relation, the *frame preserving update* $a \rightsquigarrow B$ (for $a \in \mathcal{M}$ and $B \subseteq \mathcal{M}$) by:

$$a \rightsquigarrow B \iff \forall x \in \mathcal{M}, a \cdot x \in \mathcal{V} \Rightarrow \exists b \in B, b \cdot x \in \mathcal{V}.$$

  If B is the singleton set $\{b\}$, we write $a \rightsquigarrow b$ for $a \rightsquigarrow \{b\}$.
- Introduce new *update modality* $\Rrightarrow P$, with associated frame preserving updates. Typing:

$$\frac{\Gamma \vdash P : \mathsf{Prop}}{\Gamma \vdash \Rrightarrow P : \mathsf{Prop}}$$

Update Modality

- Intuition: $\Rightarrow\!\!\Rightarrow P$ holds for a resource r, if from r we can do a frame-preserving update to some r' that satisfies P.
- Thus the update modality $\Rightarrow\!\!\Rightarrow P$ provides a way, inside the logic, to talk about the resources we *could* own after performing an update to what we *do* own.
- Laws:

$$\text{UPD-MONO}\ \frac{P \vdash Q}{\Rightarrow\!\!\Rightarrow P \vdash \Rightarrow\!\!\Rightarrow Q} \qquad \text{UPD-INTRO}\ \frac{}{P \vdash \Rightarrow\!\!\Rightarrow P} \qquad \text{UPD-IDEMP}\ \frac{}{\Rightarrow\!\!\Rightarrow\!\!\Rightarrow P \vdash \Rightarrow\!\!\Rightarrow P} \qquad \text{UPD-FRAME}\ \frac{}{P * \Rightarrow\!\!\Rightarrow Q \vdash \Rightarrow\!\!\Rightarrow (P * Q)}$$

Allocation and Ghost Update via Update Modality

- Allocation and update of ghost resources captured by the following rules:

$$\textsc{Ghost-alloc}\ \frac{a \in \mathcal{V}}{\mathsf{True} \vdash \Rrightarrow \exists \gamma.\, \boxed{a}^{\gamma}} \qquad \textsc{Ghost-update}\ \frac{a \rightsquigarrow b}{\boxed{a}^{\gamma} \vdash \Rrightarrow \boxed{b}^{\gamma}}$$

Ghost Update and Hoare Triples

- Idea: ghost state is abstract, so should be able to update it in pre- and postconditions.
- Define *view shift*

$$P \Rrightarrow Q = \Box(P \Rightarrow \models\!\Rrightarrow Q)$$

- The generalized rule of consequence is then

$$\frac{S \vdash P' \Rrightarrow P \qquad S \vdash \{P\}\, e\, \{v.Q\} \qquad S \vdash \forall v.\, Q(v) \Rrightarrow Q'(v)}{S \vdash \{P'\}\, e\, \{v.Q'\}}\ \textsc{Ht-csq}$$

Recovering the ad-hoc example ghost state

- Let M be resource algebra with carrier $\{\mathrm{S}, \mathrm{F}, \bot\}$, and with $\mathrm{F} \cdot \mathrm{F} = \mathrm{F}$ and otherwise $x \cdot y = \bot$.
- Only frame preserving update (verifty this – exercise!)

$$\mathrm{S} \rightsquigarrow \mathrm{F}$$

- Then, with the rules presented above, we can recover the rules FTOK-DUPLICABLE, S-S-INCOMPATIBLE and S-F-INCOMPATIBLE, which were postulated in the beginning.
- Moreover, the Hoare rules that were postulated earlier (HT-TOKEN-UPDATE-POST, HT-TOKEN-UPDATE-PRE, and HT-TOKEN-ALLOC) can also be derived (exercise!).

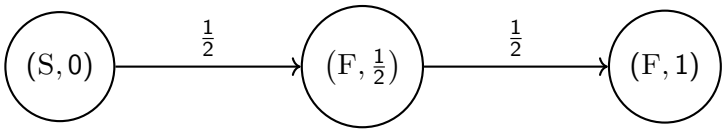
A More Precise Spec for Our Example

- We can now define an invariant, with which we can prove the following more precise specification:

$$\{\ell \hookrightarrow n\}\,(e \mid\mid e);!\,\ell\,\{v.v = n+1 \vee v = n+2\}$$

- Invariant:

$$\begin{aligned} I(\gamma_1,\gamma_2,n) = {} & \ell \hookrightarrow n * \boxed{\mathrm{S}}^{\gamma_1} \vee \\ & \ell \hookrightarrow (n+1) * \boxed{\mathrm{F}}^{\gamma_1} * \boxed{\tfrac{1}{2}}^{\gamma_2} \vee \\ & \ell \hookrightarrow (n+2) * \boxed{\mathrm{F}}^{\gamma_1} * \boxed{1}^{\gamma_2} \end{aligned}$$



- See Iris Lecture Notes for the proof.

Summary

- Ghost resources: resource algebras.
- Updates have to be *frame preserving*!
- Many constructions of resource algebras.
- The key point of verification of concrete concurrent programs often boils down to choosing the right resource algebra!