A Symbolic Decision Procedure for GKAT

Abstract

1 Symbolic Derivatives

Given a finite set K, a *symbolic Deterministic Kleene Coalgebra with Tests* (sDKCT) $\mathcal{S} \triangleq \langle S, \epsilon, \delta\rangle$ over a boolean algebra $\mathcal{B}$ consists of a state set S consists of a accepting boolean ϵ and a transition function δ:

$$\epsilon : S \to \mathcal{B}, \qquad \delta : S \to S \to K \to \mathcal{B},$$

where for all states $s \in S$, all the booleans are "disjoint"; namely the conjunction of any two expression from the set $\epsilon(s) + \{\delta(s, s', p) \mid s' \in S, p \in K\}$ are false. The ordering on $\mathcal{S}$ is defined pointwise, namely

$$\langle \epsilon_1, \delta_1\rangle \leq \langle \epsilon_2, \delta_2\rangle \text{ when } \forall s, s' \in S, \forall p \in K, \epsilon_1(s) \leq \epsilon_2(s) \text{ and } \delta_1(s, s', p) \leq \delta_2(s, s', p),$$

where the ordering on the right hand side is defined in the boolean algebra.

Intuitively, the elements of boolean is treated as the "symbolic" transitions, which can be think of as a set of labels by the classical stone's representation theorem. In particular, when the boolean algebra is the free boolean algebra BExp_B generated by a finite set B, the labels can be considered as atoms, since $\mathsf{BExp}_B \cong 2^{\mathbf{At}_B}$. Then we can explain the intuition for the definition of DKCT

- $\epsilon(s)$ can be thought of as a set of labels that is accepted by the state s.
- $\delta(s, s', p)$ denotes all the labels that can transition from s to s' while executing p.

Then the labels that is not in either operations are implicitly rejecting.

Given a sDKCT $\mathcal{S} \triangleq \langle S, \epsilon, \delta\rangle$ and a state $s \in S$, we use the notation $s \Rightarrow b$ to denote that $\epsilon(s) \geq b$, and we use the notation $s \xrightarrow{b|p} s'$ to denote that $\delta(s)_3(s', p) \geq b$, where the ordering is the typical ordering in boolean algebra.

The symbolic derivatives for GKAT forms a DKCT where the states are represented by GKAT expression $\mathsf{GKAT}_{K,B}$, and the boolean algebra is the free boolean algebra BExp_B over the test alphabet B:

$$\epsilon : \mathsf{GKAT}_{K,B} \to \mathsf{BExp}_B; \qquad \delta : \mathsf{GKAT}_{K,B} \to \mathsf{GKAT}_{K,B} \to K \to \mathsf{BExp}_B.$$

the accepting map ϵ and transition map δ can be defined by the smallest maps that satisfy the following rules:

$$\frac{}{b \Rightarrow b} \qquad \frac{}{p \xrightarrow{1|p} 1} \qquad \frac{e \Rightarrow a}{e +_b f \Rightarrow ba} \qquad \frac{f \Rightarrow a}{e +_b f \Rightarrow \bar{b}a} \qquad \frac{e \xrightarrow{a|p} e'}{e +_b f \xrightarrow{b\wedge a|p} e'} \qquad \frac{f \xrightarrow{a|p} f'}{e +_b f \xrightarrow{\bar{b}\wedge a|p} f'}$$

$$\frac{e \Rightarrow a \qquad f \Rightarrow b}{e \cdot f \Rightarrow a \wedge b} \qquad \frac{e \Rightarrow a \qquad f \xrightarrow{b|p} f'}{e \cdot f \xrightarrow{a\wedge b|p} f'} \qquad \frac{e \xrightarrow{b|p} e'}{e \cdot f \xrightarrow{b|p} e' \cdot f} \qquad \frac{}{e^{(b)} \Rightarrow \bar{b}} \qquad \frac{e \xrightarrow{a|p} e'}{e^{(b)} \xrightarrow{b\wedge a|p} e' \cdot e^{(b)}}$$

2 Implementing derivatives

Since the rules listed above is monotonic, that is if we enlarge the boolean in the premise, we also enlarge the boolean in the result. So we can simply pick the largest boolean on the premise, which is either $\epsilon(e)$ or $\delta(e, e', p)$. Since the disjunction of two boolean $a \vee b$ is the smallest boolean that is greater than both a and b, thus we can implement ϵ and δ by iteration through all the rules, and take the disjunction of all the possible booleans.

$$\begin{aligned} \epsilon(b) &\triangleq b & \epsilon(e +_b f) &\triangleq (b \wedge \epsilon(e)) \vee (\bar{b} \wedge \epsilon(f)) \\ \epsilon(q) &\triangleq 0 & \epsilon(e \cdot f) &\triangleq \epsilon(e) \wedge \epsilon(f) \\ \epsilon(e^{(b)}) &\triangleq \bar{b} \end{aligned}$$

However, the derivative function

$$\delta : \mathsf{GKAT}_{K,B} \to \mathsf{GKAT}_{K,B} \to K \to \mathsf{BExp}_B,$$

poses challenges in the implementation, since $\mathsf{GKAT}_{K,B}$ is infinite, so it will be impractical to search through all the expressions. However, we can treat the function δ extensionally:

$$\mathsf{GKAT}_{K,B} \to K \to \mathsf{BExp}_B \subseteq 2^{\mathsf{GKAT}_{K,B} \times K \times \mathsf{BExp}_B},$$

and then because the boolean expression don't overlap for each input, hence the boolean expression are necessarily unequal; we can model the set by a partial map, thus all the δ can be represented as a function to the following partial maps.

$$\begin{aligned} \delta &: \mathsf{GKAT}_{K,B} \to (\mathsf{BExp}_B \nrightarrow \mathsf{GKAT}_{K,B} \times K) \\ \delta(b) &\triangleq \{\} \\ \delta(q) &\triangleq \{1 \mapsto (1, p)\} \\ \delta(e +_b f) &\triangleq \{b \wedge a \mapsto (e', p) \mid a \mapsto (e', p) \in \delta(e)\} \cup \{\bar{b} \wedge a \mapsto (f', p) \mid a \mapsto (f', p) \in \delta(f)\} \\ \delta(e \cdot f) &\triangleq \{b \mapsto (e' \cdot f, p) \mid b \mapsto (e, p) \in \delta(e)\} \cup \{\epsilon(e) \wedge b \mapsto (f', p) \mid b \mapsto (f', p) \in \delta(f)\} \\ \delta(e^{(b)}) &\triangleq \{b \wedge a \mapsto (e' \cdot e^{(b)}, p) \mid a \mapsto (e', p) \in \delta(e)\}. \end{aligned}$$

FIXME: we still need to think about what representation do we want to go with, with this representation, you can also have duplicated K. We need a good example here of why don't we go with map. One good argument is that this representation is complex.

3 Normalized Bisimulation

Bisimilarity implies language sematnical equivalence

$$s \sim t \implies [\![s]\!] = [\![t]\!]$$

, however the converse doesn't hold unless we are working in a *normal* GKAT coalgebra. It is well known that two states in a normal GKAT coalgebra if and only if they have the same language semantics

$$s \sim t \iff [\![s]\!] = [\![t]\!].$$

However, instead of normalizing automatons, we can in fact, check for normality on the fly, this is important to design a derivatives based decision algorithm.

We will use a notion of normalized bisimulation, where we will enlarge the bisimulation by dead states, since all the dead states are trace equivalent. Operationally, when we find a discrepancy during

the bisimulation algorithm we will check whether the states causing the discrepancy are dead. We can prove the correctness of this algorithm by proving that they are equivalent to normalize and then computing bisimulation. Let G be the functor defining GKAT coalgebra:

$$G \triangleq (2 + (-) \times K)^{\mathbf{At}}.$$

Given two DKCT $\mathcal{S}, \mathcal{T}$, and a relation on these two expression $\sim \subseteq S \times T$, we can define $\overline{\sim} \subseteq (2 + S \times K) \times (2 + T \times K)$ as the smallest relation generated by the following rules: for all $p, q \in K$ and states $s' \in S$, $t' \in T$:

$$\textsc{Bisim-0}\ \frac{}{0 \mathrel{\overline{\sim}} 0} \qquad \textsc{Bisim-1}\ \frac{}{1 \mathrel{\overline{\sim}} 1} \qquad \textsc{Bisim-Trans}\ \frac{s' \sim t'}{(s', p) \mathrel{\overline{\sim}} (t', p)} \qquad \textsc{Norm-Left}\ \frac{s' \text{ is dead}}{(s', p) \mathrel{\overline{\sim}} 0} \qquad \textsc{Norm-Right}\ \frac{t' \text{ is dead}}{0 \mathrel{\overline{\sim}} (t', p)} \qquad \textsc{Norm-Both}\ \frac{s', t' \text{ are both dead}}{(s', p) \mathrel{\overline{\sim}} (t', q)}$$

Intuitively, $\overline{\sim}$ enlarges the relation $\sim$ for *results* of a transition. for example, it allows transition to dead states to equivalent to rejection, and so on. This extension enables "on-the-fly normalization", where to determine whether $s \mathrel{\overline{\sim}} t$, we can simply first check for bisimulation using the bisimulation rules; and if all failed, we can check the normalization rules.

We call the bisimulation algorithm with on-the fly normalization "*normalized bisimulation*". This concept can be summed up nicely using diagrams. For a relation $\sim$, then there is a straight forward embedding $(2 + \sim \times K) \hookrightarrow \overline{\sim}$: This embedding generates all the elements that is formed by the bisimulation rules of $\overline{\sim}$, also note that $\sim$ is not restricted to live states, but dead states as well; this means that if two dead states follows the bisimulation rules, then we don't need to fall back to the normalization rules. Since detecting whether a state is dead requires iterating through all its predecessors, we can use this property to forgo dead state detection until all the bisimulation rules failed.

The aforementioned embedding $(2 + \sim \times K) \hookrightarrow \overline{\sim}$ can be point-wise lifted to $G(\sim) \hookrightarrow (\mathbf{At} \to \overline{\sim})$, thus our normalized algorithm can be seen as finding a normalized bisimulation $\sim$ that satisfy the following diagram:

$$\begin{array}{ccccc} S & \xleftarrow{\pi_1} & \sim & \xrightarrow{\pi_2} & T \\ \downarrow{\scriptstyle \delta_{\mathcal{S}}} & & \downarrow{\scriptstyle \exists} & & \downarrow{\scriptstyle \delta_{\mathcal{T}}} \\ (2 + S \times K)^{\mathbf{At}} & \xleftarrow{{\pi_1}^{\mathbf{At}}} & (\overline{\sim}^{\mathbf{At}}) & \xrightarrow{{\pi_2}^{\mathbf{At}}} & (2 + T \times K)^{\mathbf{At}} \end{array}$$

Notice, unlike $G(\sim)$, the map $\sim \mapsto (\overline{\sim})^{\mathbf{At}}$ is not a functor, since it is only defined on relations.

Example 1. *Let's consider a example with a single primitive test $B \triangleq \{b\}$, thus atoms are defined as $\mathbf{At}_B = \{\{b\}, \emptyset\}$. We focus on two states in the bisimulation s and t. with their respective transition defined as follows*

$$\delta_{\mathcal{S}}(s, \{b\}) \triangleq 0 \qquad \delta_{\mathcal{S}}(t, \{b\}) \triangleq (t', p)$$
$$\delta_{\mathcal{S}}(s, \emptyset) \triangleq (s', p) \qquad \delta_{\mathcal{S}}(t, \emptyset) \triangleq (t', p)$$

It is quite clear s and t will not be bisimular, however, there can be a normalized bisimulation relation between them, provided that $s' \sim t'$ and t' is dead:

$$\begin{array}{ccccc} s & \xleftarrow{\pi_1} & (s,t) & \xrightarrow{\pi_2} & t \\ \downarrow{\scriptstyle \delta_{\mathcal{S}}} & & \downarrow{\scriptstyle \exists} & & \downarrow{\scriptstyle \delta_{\mathcal{T}}} \\ \{b\} \mapsto 0 & \xleftarrow{{\pi_1}^{\mathbf{At}}} & \{b\} \mapsto 0, (t', p) & \xrightarrow{{\pi_2}^{\mathbf{At}}} & \{b\} \mapsto (t', p) \\ \emptyset \mapsto (s', p) & & \emptyset \mapsto (s', p), (t', p) & & \emptyset \mapsto (t', p) \end{array}$$

Notice that $0 \mathrel{\overline{\sim}} (t', p)$ *because* t' *is dead, and* $(s', p) \mathrel{\overline{\sim}} (t', p)$ *because we assume that* $s' \sim t'$*. Furthermore, if both* s' *and* t' *are dead, we can establish* $s' \mathrel{\overline{\sim}} t'$ *using either the* Bisim-Trans *rule or* Norm-Both *rule, and we are free to pick the more efficient check in our implementation.*

Theorem 1 (Universiality)**.** *Given a normalized bisimulation relation* $\sim \subseteq S \times T$

$$\begin{array}{ccccc} S & \xleftarrow{\pi_1} & \sim & \xrightarrow{\pi_2} & T \\ \downarrow{\scriptstyle \delta_{\mathcal{S}}} & & \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle \delta_{\mathcal{T}}} \\ (2 + S \times K)^{\mathbf{At}} & \xleftarrow{{\pi_1}^{\mathbf{At}}} & (\overline{\sim})^{\mathbf{At}} & \xrightarrow{{\pi_2}^{\mathbf{At}}} & (2 + T \times K)^{\mathbf{At}} \end{array}$$

Then h *is unique, and defined as follows:*

$$h : \sim \to \overline{\sim}^{\mathbf{At}}$$
$$h(s, t, \alpha) \triangleq (\delta_{\mathcal{S}}(s, \alpha), \delta_{\mathcal{T}}(s, \alpha))$$

Proof. The commutativity can be verified by computation. We only need to show uniqueness.

The above commutative diagram implies the commutativity of the following diagram:

$$\begin{array}{ccccc} & & \sim & & \\ & \swarrow^{\delta_{\mathcal{S}} \circ \pi_1} & \downarrow{\scriptstyle h} & \searrow^{\delta_{\mathcal{T}} \circ \pi_2} & \\ & & (\overline{\sim})^{\mathbf{At}} & & \\ & & \downarrow{\scriptstyle i} & & \\ (2 + S \times K)^{\mathbf{At}} & \xleftarrow[\pi_1^{\mathbf{At}}]{} & (2 + S \times K)^{\mathbf{At}} \times (2 + T \times K)^{\mathbf{At}} & \xrightarrow[\pi_2^{\mathbf{At}}]{} & (2 + T \times K)^{\mathbf{At}} \end{array}$$

Since $(-)^{\mathbf{At}}$ is a right adjoint and right adjoints preserve limits, $(2 + S \times K)^{\mathbf{At}} \times (2 + T \times K)^{\mathbf{At}}$ is the product of $(2 + T \times K)^{\mathbf{At}}$ and $(2 + T \times K)^{\mathbf{At}}$. Hence, $h \circ i$ is the unique function to make the above diagram commute. Assume we can replace h with g while preserving the commutativity of the diagram:

$$\begin{array}{ll} i \circ h = i \circ g & (2 + S \times K)^{\mathbf{At}} \times (2 + T \times K)^{\mathbf{At}} \text{ is a product.} \\ \Longrightarrow h = g & i \text{ is a embedding, hence left cancelable.} \end{array}$$

□

Definition 1. *Given a DKCT* $\mathcal{S} = \langle S, \delta \rangle$*, a* sub-DKCT $\mathcal{S}' = \langle S', \delta|_{S'} \rangle$ *is a coagebra induced by* $S' \subseteq S$*, a sub-DKCT can be characterized as a pullback square:*

$$\begin{array}{ccc} S' & \xhookrightarrow{i} & S \\ \downarrow{\scriptstyle \delta|_{S'}} & & \downarrow{\scriptstyle \delta} \\ G(S') & \xhookrightarrow{i} & G(S) \end{array}$$

By the commutativity of this diagram i *is also a DKCT homomorphism. A* principle sub-DKCT *of state* $s \in S$*, denoted* $\langle s \rangle$ *is the smallest sub-DKCT that contains* s*. We can show that the principle DKCT of* s *exactly characterize the DKCT induced by the reachable state of* s*.*

Lemma 1. *A state* s *is live if and only if there exists a accepting state in* $\langle s \rangle$*; and a state* s *is dead if and only if there is no accepting state in* $\langle s \rangle$*.*

Corollary 1. s *is dead if and only if* $\langle s \rangle$ *is all dead.*

Proof. Take any state $s' \in \langle s \rangle$, we can construct the DKCT $\langle s' \rangle$. Since $\langle s' \rangle$ is the smallest sub-DKCT that contains s', and $s' \in \langle s \rangle$, thus $\langle s' \rangle \subseteq \langle s \rangle$. Finally, because there is no accepting state in $\langle s \rangle$, there cannot be any accepting state in $\langle s' \rangle$ thus s' is also dead. □

In fact the normalization operation commutes with principle sub-DKCT. This result is intuitive as taking all the reachable states and then normalize is the same as normalize and then take all the reachable state.

Lemma 2. *Given two sub-DKCT $\mathcal{S}_1 \sqsubseteq \mathcal{S}$ and $\mathcal{S}_2 \sqsubseteq \mathcal{T}$, then*

$$\mathcal{S}_1 \sqsubseteq \mathcal{S}_2 \iff \mathcal{S}_1 \subseteq \mathcal{S}_2$$

Proof. because their transition function are both restriction of $\delta_{\mathcal{S}}$. □

Lemma 3. *Homomorphism preserves sub-DKCT, given $h : \mathcal{S} \to \mathcal{T}$:*

$$\mathcal{S}' \sqsubseteq \mathcal{S} \implies h(\mathcal{S}') \sqsubseteq h(\mathcal{S}) \sqsubseteq \mathcal{T}$$

Proof. consequence of previous lemma. □

Lemma 4. *Homomorphism preserves principle sub-DKCT. Specifically, given a homomorphism $h : \mathcal{S} \to \mathcal{T}$,*

$$h(\langle s \rangle_{\mathcal{S}}) = \langle h(s) \rangle_{\mathcal{T}},$$

where $h(\langle s \rangle_{\mathcal{S}})$ is the image of $\langle s \rangle_{\mathcal{S}}$ under h.

Proof. Specifically, we will need to show that $h(\langle s \rangle_{\mathcal{S}})$ is the smallest sub-DKCT of $\mathcal{T}$ that contain $h(s)$. First by definition of image, $h(s) \in h(\langle s \rangle_{\mathcal{S}})$, second because h is a homomorphism $\mathcal{S} \to \mathcal{T}$, and $\langle s \rangle_{\mathcal{S}} \sqsubseteq \mathcal{S}$, thus $h(\langle s \rangle_{\mathcal{S}}) \sqsubseteq \mathcal{T}$.

Finally, take any $\mathcal{T}' \sqsubseteq \mathcal{T}$, and $h(s) \in \mathcal{T}'$. We take the preimage of $\mathcal{T}'$ under h, since $s \in h^{-1}(\mathcal{T})$, thus $\langle s \rangle_{\mathcal{S}} \sqsubseteq h^{-1}(\mathcal{T})$, which implies $h(\langle s \rangle_{\mathcal{S}}) \sqsubseteq \mathcal{T}$. □

Corollary 2. *Homomorphism preserves live/dead/accepting states. Given a homomorphism $h : \mathcal{S} \to \mathcal{T}$ then $s \in \mathcal{S}$ is a live/dead/accepting state if and only if $h(s) \in \mathcal{T}$ is a live/dead/accepting state, respectively.*

Proof. First, homomorphism preserves accepting state by definition. Then because homomorphism preserves principle sub-DKCT and a state s is live if and only if there exists an accepting state $\langle s \rangle$, thus homomorphism will also preserve the liveness of s. □

Corollary 3. *sub-DKCT preserves principle sub-DKCT, that is, given $\mathcal{S}' \sqsubseteq \mathcal{S}$ and $s \in \mathcal{S}'$, then*

$$\langle s \rangle_{\mathcal{S}'} = \langle s \rangle_{\mathcal{S}}.$$

Proof. Because inclusion $i : \mathcal{S}' \to \mathcal{S}$ is a homomorphism, thus

$$\langle s \rangle_{\mathcal{S}'} = i(\langle s \rangle_{\mathcal{S}'}) = \langle i(s) \rangle_{\mathcal{S}} = \langle s \rangle_{\mathcal{S}}.$$ □

Corollary 4. *sub-DKCT preserves liveness, that is given $\mathcal{S}' \sqsubseteq \mathcal{S}$ and $s \in \mathcal{S}'$,*

$$s \text{ is live in } \mathcal{S}' \iff s \text{ is live in } \mathcal{S}$$

Lemma 5. norm *is a functor, and it preserves inclusion, that is*

$$\mathcal{S} \sqsubseteq \mathcal{T} \implies \mathrm{norm}(\mathcal{S}) \sqsubseteq \mathrm{norm}(\mathcal{T}).$$

Lemma 6. *Given a DKCT $\mathcal{S}$, and two sub-DKCT $\mathcal{S}_1 \sqsubseteq \mathcal{S}$ and $\mathcal{S}_2 \sqsubseteq \mathcal{S}$, $\mathcal{S}_1 = \mathcal{S}_2$ if and only if their carrier set is equal.*

Lemma 7. *Given a DKCT $\mathcal{S}$, its dead states induces a sub-DKCT.*

Theorem 2. *Given a DKCT $\mathcal{S}$ and a live state $s \in \mathcal{S}$:*

$$\langle s \rangle_{\mathrm{norm}(\mathcal{S})} = \mathrm{norm}(\langle s \rangle_{\mathcal{S}}).$$

Proof. We need to show $\mathrm{norm}(\langle s \rangle_{\mathcal{S}})$ is the smallest sub-DKCT in $\mathrm{norm}(\mathcal{S})$ that contains s.

First, $\mathrm{norm}(\langle s \rangle_{\mathcal{S}})$ contains s because s is live. Second, take any $\mathcal{T} \sqsubseteq \mathrm{norm}(\mathcal{S})$, we will use $\overline{\mathcal{T}}$ to denote the smallest sub-DKCT of $\mathcal{S}$ that contains all the states in $\mathcal{T}$. Since $\mathcal{T}$ is a sub-DKCT of $\mathrm{norm}(\mathcal{S})$, $\overline{\mathcal{T}} \setminus \mathcal{T}$ can only contain dead states, thus $\mathrm{norm}(\overline{\mathcal{T}}) = \mathcal{T}$.

$$\begin{aligned}
& \langle s \rangle_{\mathcal{S}} \sqsubseteq \overline{\mathcal{T}} && \overline{\mathcal{T}} \text{ contains } s \\
\implies & \mathrm{norm}(\langle s \rangle_{\mathcal{S}}) \sqsubseteq \mathrm{norm}(\overline{\mathcal{T}}) && \text{monotonicity of norm} \\
\implies & \mathrm{norm}(\langle s \rangle_{\mathcal{S}}) \sqsubseteq \mathcal{T} && \mathrm{norm}(\overline{\mathcal{T}}) = \mathcal{T}
\end{aligned}$$

□

We will show this by establishing the equivalence between "normalized bisimulation" and "bisimulation in normalized coalgebra". Given a GKAT coalgebra $\mathcal{S}$, we can normalize it via removing the dead states and reroute all the transitions to dead state as direct rejections. Similarly, any relation $\sim \subseteq \mathcal{S} \times \mathcal{T}$ can be normalized by filtering out all the states that are live:

$$\mathrm{norm}(\sim) \triangleq \{(s,t) \mid \text{both } s \text{ and } t \text{ are live states}\}.$$

Thus there is a natural embedding $\mathrm{norm}(\sim) \hookrightarrow \sim$ by inclusion. Similarly, there is a embedding $G(\mathrm{norm}(\sim)) \to (\overline{\sim})^{\mathbf{At}}$ by lifting the following embedding point-wise via $(-)^{\mathbf{At}}$:

$$\begin{aligned}
2 + \mathrm{norm}(\sim) \times K &\hookrightarrow \overline{\sim} \\
0 &\mapsto (0,0) \\
1 &\mapsto (1,1) \\
(s,t), p &\mapsto (s,p), (t,p)
\end{aligned}$$

These two embeddings allows us to construct the following pullback square:

$$\begin{array}{ccc}
\mathrm{norm}(\sim) & \xhookrightarrow{i} & \sim \\
{\scriptstyle \mathrm{norm}(h)}\downarrow & & \downarrow{\scriptstyle h} \\
G(\mathrm{norm}(\sim)) & \xhookrightarrow{i} & (\overline{\sim})^{\mathbf{At}}
\end{array}$$

Theorem 3. *A normalized bisimulation induces a bisimulation in the normal coalgebras. That is*

$$\begin{array}{ccccc}
S & \xleftarrow{\pi_1} & \sim & \xrightarrow{\pi_2} & T \\
{\scriptstyle \delta_{\mathcal{S}}}\downarrow & & {\scriptstyle (\delta_{\mathcal{S}}, \delta_{\mathcal{T}})}\downarrow & & \downarrow{\scriptstyle \delta_{\mathcal{T}}} \\
G(S) & \xleftarrow{{\pi_1}^{\mathbf{At}}} & \overline{\sim}^{\mathbf{At}} & \xrightarrow{{\pi_2}^{\mathbf{At}}} & G(T)
\end{array}
\implies
\begin{array}{ccccc}
\mathrm{norm}(S) & \xleftarrow{\pi_1} & \mathrm{norm}(\sim) & \xrightarrow{\pi_2} & \mathrm{norm}(T) \\
{\scriptstyle \delta_{\mathrm{norm}(\mathcal{S})}}\downarrow & & {\scriptstyle \mathrm{norm}(h)}\downarrow & & \downarrow{\scriptstyle \delta_{\mathrm{norm}(\mathcal{T})}} \\
G(\mathrm{norm}(S)) & \xleftarrow{G(\pi_1)} & G(\mathrm{norm}(\sim)) & \xrightarrow{G(\pi_2)} & G(\mathrm{norm}(T))
\end{array}$$

Proof. By computation. □

Lemma 8. *Given a normalized bisimulation $\sim \subseteq \mathcal{S} \times \mathcal{T}$, and a pair of state $s \sim t$, then the range of $\pi_1 : \sim \to S$, will cover all the live states in $\langle s \rangle$; similarly for the state $t \in \mathcal{T}$.*

Proof. let $\text{norm}(\sim)|_{\langle s\rangle}$ be all the live elements of $\sim$ restricted to $\langle s\rangle$:

$$\text{norm}(\sim)|_{\langle s\rangle} \triangleq \{(s', t') \in \sim \mid s' \in \langle s\rangle, \text{both } s' \text{ and } t' \text{ are live}\}.$$

since all the live states in $\langle s\rangle$ is the carrier of $\text{norm}(\langle s\rangle)$, we only need to show $\pi_1 : \text{norm}(\sim)|_{\langle s\rangle} \to \langle s\rangle$ is surjective.

Since $\sim$ is a normalize bi-simulation, then $\text{norm}(\sim)$ is a bisimulation, and the following diagram commute by computation:

$$\begin{array}{ccc} \text{norm}(\sim)|_{\langle s\rangle} & \xrightarrow{\pi_1} & \text{norm}(\langle s\rangle) \\ \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle \delta_{\text{norm}(\mathcal{S})}} \\ G(\text{norm}(\sim)|_{\langle s\rangle}) & \xrightarrow{G(\pi_1)} & G(\text{norm}(\langle s\rangle)) \end{array}$$

Because of above diagram, $\mathbf{Img}(\pi_1) \subseteq \text{norm}(\langle s\rangle)$, the range of π_1, also forms a sub-DKCT of $\text{norm}(\mathcal{S})$ with the transition: $\delta_{\text{norm}(\mathcal{S})} : \mathbf{Img}(\pi_1) \to G(\mathbf{Img}(\pi_1))$ then there are two cases:

- if s is dead, then $\text{norm}(\langle s\rangle)$ is empty, thus π_1 is surjective,
- if s is live, then $\text{norm}(\langle s\rangle) = \langle s\rangle_{\text{norm}(\mathcal{S})}$, because $\mathbf{Img}(\pi_1)$ is a sub-DKCT and contains s, thus

  $$\text{norm}(\langle s\rangle) = \langle s\rangle_{\text{norm}(\mathcal{S})} \sqsubseteq \mathbf{Img}(\pi_1).$$

  Hence $\mathbf{Img}(\pi_1)$ covers $\text{norm}(\langle s\rangle)$ and π_1 is surjective.

□

Theorem 4. *Given a dead state s and a live state t, there doesn't exists a normalized bisimulation $\sim$ s.t. $s \sim t$ or $t \sim s$.*

Proof. WLOG we will only prove that $s \sim t$ will lead to a contradiction. We can restrict the simulation diagram of $\sim$ to $\langle s\rangle$ and $\text{norm}(\langle t\rangle)$, let

$$\sim_{s,\hat{t}} = \{s', t' \mid s' \sim t', s' \in \langle s\rangle, t' \in \text{norm}(\langle t\rangle)\}.$$

Note that only $\mathcal{T}$ is restricted to $\text{norm}(\langle t\rangle)$; while $\mathcal{S}$ is restricted to $\langle s\rangle$, without norm. This discrepancy is because s is dead, thus $\text{norm}(\langle s\rangle)$ is empty. If we restrict $\sim$ to $\text{norm}(\langle t\rangle) \times \text{norm}(\langle s\rangle)$, then there will be no element in the restriction, since $\text{norm}(\langle t\rangle) \times \text{norm}(\langle s\rangle)$ is empty.

We can show that $\sim_{s,\hat{t}}$ form a normalized bisimulation between $\langle s\rangle$ and $\text{norm}(\langle t\rangle)$.

$$\begin{array}{ccccc} \langle s\rangle & \xleftarrow{\pi_1} & \sim_{s,\hat{t}} & \xrightarrow{\pi_2} & \text{norm}(\langle t\rangle) \\ \downarrow{\scriptstyle \delta_{\mathcal{S}}} & & \downarrow{\scriptstyle (\delta_{\mathcal{S}}, \text{norm}(\delta_{\mathcal{T}}))} & & \downarrow{\scriptstyle \text{norm}(\delta_{\mathcal{T}})} \\ G(\langle s\rangle) & \xleftarrow{\pi_1^{\mathbf{At}}} & \overline{\sim_{s,\hat{t}}}^{\mathbf{At}} & \xrightarrow{\pi_2^{\mathbf{At}}} & G(\text{norm}(\langle t\rangle)) \end{array}$$

Then we can perform epi-mono factorization on the right square to obtain the range of π_2:

$$\begin{array}{ccccc} \sim_{s,\hat{t}} & \xrightarrow{\pi_2} & \mathbf{Img}(\pi_2) & \xhookrightarrow{i} & \text{norm}(\langle t\rangle) \\ \downarrow{\scriptstyle (\delta_{\mathcal{S}}, \text{norm}(\delta_{\mathcal{T}}))} & & \downarrow{\scriptstyle \text{norm}(\delta_{\mathcal{T}})} & & \downarrow{\scriptstyle \text{norm}(\delta_{\mathcal{T}})} \\ \overline{\sim_{s,\hat{t}}}^{\mathbf{At}} & \xrightarrow{\pi_2^{\mathbf{At}}} & G(\mathbf{Img}(\pi_2)) & \xhookrightarrow{G(i)} & G(\text{norm}(\langle t\rangle)) \end{array}$$

Thus,

$$\mathbf{Img}(\pi_2) \sqsubseteq \text{norm}(\langle t\rangle) \sqsubseteq \text{norm}(\mathcal{T}).$$

Recall that $\mathrm{norm}(\langle t\rangle_{\mathcal{T}}) = \langle t\rangle_{\mathrm{norm}(\mathcal{T})}$, hence $\mathrm{norm}(\langle t\rangle)$ is the smallest sub-DKCT of $\mathrm{norm}(\mathcal{T})$, since $s \sim t$, thus $t \in \mathbf{Img}(\pi_2)$. Hence $\mathbf{Img}(\pi_2) \sqsupseteq \mathrm{norm}(\langle t\rangle)$, thus π_2 is surjective.

Since t is a live state, hence there exists an accepting state t' in $\mathrm{norm}(\langle t\rangle)$; because s is a dead state, thus all the states in $\langle s\rangle$ is dead. Because $\pi_2 : \sim_{s,\hat{t}} \to T$ is surjective, there exists a dead state $s' \in \langle s\rangle$ s.t. $s' \sim t'$. However, a dead state cannot be normally bisimilar to a accepting state, since $((s', p), 1) \notin \overline{\sim_{s,\hat{t}}}$ for all dead state s' and action p. □

Theorem 5. *Homomorphism reflects normalized bisimulation. Let $h : \mathcal{S}_1 \to \mathcal{S}_2$, and a normalized bisimulation $\sim \subseteq \mathcal{S}_2 \times \mathcal{T}$, then there exists a normalized bisimulation $\sim_h : \mathcal{S}_1 \times \mathcal{T}$. And similarly for $\sim \subseteq \mathcal{T} \times \mathcal{S}_2$.*

Proof. $\sim_h$ can be defined as follows:

$$\sim_h \triangleq \{(s, t) \mid h(s) \sim t\}.$$

Then the function $h \times id : \sim_h \to \sim$, can be lifted to $\overline{\sim_h}^{\mathbf{At}} \to \overline{\sim}^{\mathbf{At}}$:

$$\begin{array}{ccc} \sim & \xleftarrow{h\times id} & \sim_h \\ {\scriptstyle (\delta_{\mathcal{S}_2},\delta_{\mathcal{T}})}\downarrow & & \downarrow{\scriptstyle (\delta_{\mathcal{S}_1},\delta_{\mathcal{T}})} \\ \overline{\sim}^{\mathbf{At}} & \xleftarrow[((id+h\times id)\times id)^{\mathbf{At}}]{} & \overline{\sim_h}^{\mathbf{At}} \end{array}$$

The generated function $(id + h \times id) \times id : \overline{\sim_h} \to \overline{\sim}$ is indeed well-defined by case analysis on the input in $\sim_h$, and also by the fact that homomorphism like h preserves liveness. □

Definition 2. *A guarded language L over alphabet K, B is deterministic when given a list of atoms $\alpha_0, \alpha_1, \alpha_2, ..., \alpha_n$, there is at most one word in the language $w \in L$ that has the form:*

$$\alpha_0 p_1 \alpha_1 ... p_n \alpha_n \text{ for some } p_i \in K$$

Theorem 6. *the set of deterministic guarded language over the alphabet K, B: $\mathcal{G}_{K,B}$ is a DKCT; and the trace semantics $[\![-]\!] : \mathcal{S} \to \mathcal{G}$ is a homomorphism.*

Theorem 7 (soundness and completeness)**.** *the trace of two state $s \in \mathcal{S}$ and $t \in \mathcal{T}$ are trace equivalent if and only if there exists a normalized bisimulation between them*

$$[\![s]\!] = [\![t]\!] \iff \text{there exists a normalized bisimulation } \sim \text{ s.t. } s \sim t.$$

Proof. The $\Longleftarrow$ direction can be shown by case analysis on the liveness of s and t:

- If both s and t are live,

$$s \sim t \Longrightarrow (s, t) \in \mathrm{norm}(\sim) \Longrightarrow [\![s]\!] = [\![t]\!].$$

- If one of s and t is live, and the other is dead, then $s \sim t$ cannot hold.
- If both s and t are dead, since dead states don't have any trace:

$$[\![s]\!] = [\![t]\!] = \emptyset.$$

$\Longrightarrow$ direction can be shown by constructing the identity normalized bisimulation on languages: $\sim_{id} \subseteq \mathcal{G} \times \mathcal{G}$ and then reflect both side by the language semantics $[\![-]\!] : \mathcal{S} \to \mathcal{G}$ and $[\![-]\!] : \mathcal{T} \to \mathcal{G}$. Thus we obtain a normalized bisimulation on $\mathcal{S}$ and $\mathcal{T}$. □

Corollary 5. *The language equivalence is the maximal normalized bisimulation.*

Corollary 6. *There exists a normalized bisimulation s.t. $s \sim t$ if and only if there exists a normalized bisimulation equivalence $\equiv$ s.t. $s \equiv t$.*

Proof. take the $\equiv$ to be language equivalence. □

Corollary 7. *The trace semantics of a state is preserved in sub-DKCTs, that is given $\mathcal{S}' \sqsubseteq \mathcal{S}$ and a state $s \in \mathcal{S}'$, $[\![s]\!]_{\mathcal{S}} = [\![s]\!]_{\mathcal{S}'}$*

Proof. We only need to exhibit a normalized bisimulation between $\mathcal{S}'$ and $\mathcal{S}$:

$$s \sim t \text{ when } s = t \text{ or } s \text{ and } t \text{ are both dead.}$$

To show that the relation $\sim$ is a normalized bisimulation, we only need to check the commutativity of the following diagram:

$$\begin{array}{ccccc} \mathcal{S}' & \xleftarrow{\pi_1} & \sim & \xrightarrow{\pi_2} & \mathcal{S} \\ \downarrow{\scriptstyle \delta_{\mathcal{S}}} & & \downarrow{\scriptstyle (\delta_{\mathcal{S}}, \delta_{\mathcal{S}})} & & \downarrow{\scriptstyle \delta_{\mathcal{S}}} \\ G(\mathcal{S}') & \xleftarrow{{\pi_1}^{\mathbf{At}}} & \overline{\sim}^{\mathbf{At}} & \xrightarrow{{\pi_2}^{\mathbf{At}}} & G(\mathcal{S}) \end{array}$$

The commutativity is apparent by computation. The function $(\delta_{\mathcal{S}}, \delta_{\mathcal{S}}) : \sim \to \overline{\sim}^{\mathbf{At}}$ is well-defined, by case analysis on the input:

- if $s = t$, then $\delta_{\mathcal{S}}(s, \alpha) = \delta_{\mathcal{S}}(t, \alpha)$, hence will be in $\overline{\sim}$ for all α.
- if both s and t are dead, then for all $\alpha \in \mathbf{At}$ both $\delta_{\mathcal{S}}(s, \alpha)$ and $\delta_{\mathcal{S}}(t, \alpha)$ are will either transition to a dead state or reject, thus the results will also be in $\overline{\sim}$.

□

Since sub-DKCT preserves semantics, we only need to exhibit a normalized bisimulation $\sim \subseteq \langle s \rangle \times \langle t \rangle$, to shrink the size of the bisimulation:

$$\begin{array}{ccccc} \langle s \rangle & \xleftarrow{\pi_1} & \sim_{s,t} & \xrightarrow{\pi_2} & \langle t \rangle \\ \downarrow{\scriptstyle \delta_{\mathcal{S}}} & & \downarrow{\scriptstyle (\delta_{\mathcal{S}}, \delta_{\mathcal{T}})} & & \downarrow{\scriptstyle \delta_{\mathcal{T}}} \\ G(\langle s \rangle) & \xleftarrow{{\pi_1}^{\mathbf{At}}} & \overline{\sim_{s,t}}^{\mathbf{At}} & \xrightarrow{{\pi_2}^{\mathbf{At}}} & G(\langle t \rangle) \end{array}$$

The because sub-DKCT preserves trace semantics, thus the completeness result still holds:

$$[\![s]\!]_{\mathcal{S}} = [\![t]\!]_{\mathcal{T}} \iff [\![s]\!]_{\langle s \rangle} = [\![t]\!]_{\langle t \rangle} \iff \text{there exists } \sim \subseteq \langle s \rangle \times \langle t \rangle \text{ such that } s \sim t.$$

Finally, to construct a normalized bisimulation using programs, we will need the following inductive construction theorem

Theorem 8 (Inductive Construction)**.** *Given two DKCT $\mathcal{S}$ and $\mathcal{T}$, and two of their elements $s \in \mathcal{S}$ and $t \in \mathcal{T}$, there exists a normalized bisimulation $s \sim t$ between $\langle s \rangle$ and $\langle t \rangle$, if and only if either both s and t are dead or for all $\alpha \in \mathbf{At}$, all of the following holds:*

1. *$\delta_{\mathcal{S}}(s, \alpha) = \texttt{Accept} \iff \delta_{\mathcal{T}}(t, \alpha) = \texttt{Accept}$;*
2. *If $\delta_{\mathcal{S}}(s, \alpha) = (s', p)$ and $\delta_{\mathcal{T}}(t, \alpha) = (t', p)$, then and there exists a normalized bisimulation $\sim_{s',t'}$ on $\langle s' \rangle$ and $\langle t' \rangle$, s.t. $s' \sim_{s',t'} t'$;*
3. *If $\delta_{\mathcal{S}}(s, \alpha) = (s', p)$ and $\delta_{\mathcal{T}}(t, \alpha) = (t', q)$, s.t. $p \neq q$, then both s' and t' are dead;*

4. *s reject α or transition to a dead state via α if and only if t rejects α or transition to a dead state via α.*

Proof. We first prove the $\Longrightarrow$ direction: if there exists a normalized bisimulation $\sim$, then for all $\alpha \in \mathbf{At}$, $\delta_{\mathcal{S}}(s,\alpha) \overline{\sim} \delta_{\mathcal{T}}(t,\alpha)$. And by unfolding the definition of $\overline{\sim}$, we can obtain all the conditions we desired.

For the $\Longleftarrow$ direction, we can explicitly construct $\sim$ as the union of all the normalized bisimulations $\sim_{s',t'} \subseteq \langle s' \rangle \times \langle t' \rangle$:

$$\sim = \{(s,t)\} \cup \{\sim_{s',t'} \mid \exists \alpha \in \mathbf{At}, p \in K, \delta_{\mathcal{S}}(s,\alpha) = (s',p) \text{ and } \delta_{\mathcal{T}}(t,\alpha) = (t',p)\}.$$

Then, we will need to show $\sim$ is indeed a normalized bisimulation on $\langle s \rangle$ and $\langle t \rangle$, that is for all $s' \in \langle s \rangle$ and $t' \in \langle t \rangle$:

$$s' \sim t' \Longrightarrow \delta_{\mathcal{S}}(s',\alpha) \overline{\sim} \delta_{\mathcal{T}}(t',\alpha).$$

We only need to show the above implication for $s \sim t$, as other element is already in some normalized bisimulation between sub-DKCTs of $\mathcal{S}$ and $\mathcal{T}$. And the implication $s \sim t \Longrightarrow \delta_{\mathcal{S}}(s',\alpha) \overline{\sim} \delta_{\mathcal{T}}(t',\alpha)$ can be proven by unfolding the definition of $\overline{\sim}$. □

Given a state $s \in \mathcal{S}$ we can run a simple depth-first search in $\langle s \rangle$ to determine whether s is dead. Specifically recall that s is live if and only if there exists an accepting state in s, and if s is dead, then all of $\langle s \rangle$ is dead

```
(** return whether the state is accepting*)
let is_accept (s: state): bool =
    ∃α ∈ At, δ(s, α) = Accept

(** a mutable set to keep track of states
that has been explored by check_dead function*)
let explored = ∅
(** return a all the states in ⟨s⟩ if s is dead,
otherwise return None*)
let check_dead (s: state): state set option =
    (* ⟨s⟩ has all be explored *)
    if s ∈ explored then ∅

    (* exploring ⟨s⟩ *)
    else for α ∈ At:
        if δ(s, α) = Accept then None
        else if δ(s, α) = (s', p) then
            if check_dead s' = None then None
            else explored := explored ∪ check_dead s'
        (* δ(s, α) = Reject case, skip *)

    (* finished exploring ⟨s⟩ *)
    explored ∪ {s}
```

To achieve better efficiency, we can cache all the dead state we found:

```
(** a mutable set to keep track of all the found dead states *)
let dead_states = ∅

(** a function to check whether a state is dead *)
let is_dead (s: state): bool =
    if s ∈ dead_states then true
    else if check_dead s = None then false
    else
        dead_states := dead_states ∪ check_dead s
        return true
```

Thus our algorithm can be directly obtained from previous theorem, with couple modifications

- we will finding normalized bisimulation equivalence $\sim$ in a DKCT with transition function δ, as oppose to in $\mathcal{S}$ and $\mathcal{T}$. This approach is general enough: when given $\mathcal{S}$ and $\mathcal{T}$, we can merge them via coproduct $\mathcal{S} + \mathcal{T}$, where both $\mathcal{S}$ and $\mathcal{T}$ are sub-DKCT of $\mathcal{S} + \mathcal{T}$, thus preserving the semantics of all the states.
- we will find a bisimulation equivalence instead of bisimulation relation, this allows us to use union-find structure to keep track of related states. In particular, the `union` function will mark two states as equivalent, and `eq` function will return whether two states has the same representative in the union-find.

```
(** computing the normalized bisimulation equivalence using union find *)
let equiv (s: state) (t: state): bool =
    (* if s and t are already in the normalized bisimulation equivalence *)
    if eq s t then true else

    (* checking if they are both dead *)
    if s ∈ dead_states then is_dead t else
    if t ∈ dead_states then is_dead s else

    (* all of the following condition needs to be true for all atoms *)
    forall (α ∈ At):
        match δ(s, α), δ(t, α) with
            Accept, Accept -> true
            (s', p), (t', q) ->
                if p = q
                    (* mark s and t as bisimular and check the rest*)
                    then union s t; equiv s' t'
                    else is_dead s' && is_dead t'
            (s', p), Reject -> is_dead s'
            Reject, (t', p) -> is_dead t'
            Reject, Reject -> true
            _ -> false
```

TODO: I think I still need to use one of those algorithm package...

The correctness of this algorithm can be exhibited by the following induction invariant:

$$\texttt{equiv s t} \iff \text{there exists a normalized bisimulation equivalence } \sim \text{ s.t. } s \sim t$$

Then by the soundness and completeness of normalized bisimulation:

$$\texttt{equiv s t} \iff [\![s]\!] = [\![t]\!].$$