Program Verification: Lecture 24

José Meseguer

University of Illinois at Urbana-Champaign

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$,

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$,

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities.

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_{\mathbb{C}_{\mathcal{R}}}$ has the form: $\_{\mathbb{C}_{\mathcal{R}}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$,

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_{\mathbb{C}_{\mathcal{R}}}$ has the form: $\_{\mathbb{C}_{\mathcal{R}}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$, with $[\![u \mid \varphi]\!]$ the computable subset:

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_{\mathbb{C}_{\mathcal{R}}}$ has the form: $\_{\mathbb{C}_{\mathcal{R}}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$, with $[\![u \mid \varphi]\!]$ the computable subset:

$$[\![u \mid \varphi]\!] = \{[v] \in C_{\Sigma/\vec{E},B,St} \mid \exists \rho \; s.t. \; v =_B u\rho \;\wedge\; E \cup B \vdash \varphi\rho\} \subseteq C_{\Sigma/\vec{E},B,St}$$

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_\mathbb{C}_{\mathcal{R}}$ has the form: $\_\mathbb{C}_{\mathcal{R}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$, with $[\![u \mid \varphi]\!]$ the computable subset:

$$[\![u \mid \varphi]\!] = \{[v] \in C_{\Sigma/\vec{E},B,St} \mid \exists \rho \; s.t. \; v =_B u\rho \;\wedge\; E \cup B \vdash \varphi\rho\} \subseteq C_{\Sigma/\vec{E},B,St}$$

For narrowing search we first focus on topmost rewrite theories of the form $\mathcal{R} = (\Omega, B, R)$ and choose as our Π the set of constructor patterns $u \in T_\Omega(X)_{St}$.

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_{\mathbb{C}_{\mathcal{R}}}$ has the form: $\_{\mathbb{C}_{\mathcal{R}}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$, with $[\![u \mid \varphi]\!]$ the computable subset:

$$[\![u \mid \varphi]\!] = \{[v] \in C_{\Sigma/\vec{E},B,St} \mid \exists \rho \; s.t. \; v =_B u\rho \;\wedge\; E \cup B \vdash \varphi\rho\} \subseteq C_{\Sigma/\vec{E},B,St}$$

For narrowing search we first focus on topmost rewrite theories of the form $\mathcal{R} = (\Omega, B, R)$ and choose as our Π the set of constructor patterns $u \in T_{\Omega}(X)_{St}$. A constructor pattern u coincides with the constrained constructor pattern $u|\top$.

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_{\mathbb{C}_{\mathcal{R}}}$ has the form: $\_{\mathbb{C}_{\mathcal{R}}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$, with $[\![u \mid \varphi]\!]$ the computable subset:

$$[\![u \mid \varphi]\!] = \{[v] \in C_{\Sigma/\vec{E},B,St} \mid \exists \rho \; s.t. \; v =_B u\rho \; \wedge \; E \cup B \vdash \varphi\rho\} \subseteq C_{\Sigma/\vec{E},B,St}$$

For narrowing search we first focus on topmost rewrite theories of the form $\mathcal{R} = (\Omega, B, R)$ and choose as our Π the set of constructor patterns $u \in T_{\Omega}(X)_{St}$. A constructor pattern u coincides with the constrained constructor pattern $u|\top$. The meaning function is:

Constructor Pattern Predicates

Recall from Lecture 18 that for an executable rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ with constructor subsignature Ω and state sort St, an expressive set Π of state predicate names to specify modal properties of $\mathbb{C}_{\mathcal{R}}$ is the set of constrained constructor patterns $u|\varphi$, with u a constructor Ω-term of sort St, $vars(u) = \vec{x}$, and $\varphi(\vec{x})$ a conjunction of Σ-equalities. The meaning function $\_{}_{\mathbb{C}_{\mathcal{R}}}$ has the form: $\_{}_{\mathbb{C}_{\mathcal{R}}} : (u|\varphi) \mapsto [\![u \mid \varphi]\!]$, with $[\![u \mid \varphi]\!]$ the computable subset:

$$[\![u \mid \varphi]\!] = \{[v] \in C_{\Sigma/\vec{E},B,St} \mid \exists\rho \; s.t. \; v =_B u\rho \;\wedge\; E \cup B \vdash \varphi\rho\} \subseteq C_{\Sigma/\vec{E},B,St}$$

For narrowing search we first focus on topmost rewrite theories of the form $\mathcal{R} = (\Omega, B, R)$ and choose as our Π the set of constructor patterns $u \in T_{\Omega}(X)_{St}$. A constructor pattern u coincides with the constrained constructor pattern $u|\top$. The meaning function is:

$$\_{}_{\mathbb{C}_{\mathcal{R}}} : u \mapsto [\![u]\!] =_{def} \{[v] \in T_{\Omega/B,St} \mid \exists\rho \; s.t. \; v =_B u\rho\} \subseteq T_{\Omega/B,St}.$$

Positive Constructor Pattern Formulas

Positive constructor pattern formulas $PCPattF$ have the grammar:

$$u \mid p \vee p' \mid p \wedge p' \mid \bot$$

Positive Constructor Pattern Formulas

Positive constructor pattern formulas $PCPattF$ have the grammar:

$$u \mid p \vee p' \mid p \wedge p' \mid \bot$$

where $u \in T_{\Omega}(X)_{St}$ and $p, p' \in PCPattF$.

Positive Constructor Pattern Formulas

Positive constructor pattern formulas *PCPattF* have the grammar:

$$u \mid p \vee p' \mid p \wedge p' \mid \bot$$

where $u \in T_{\Omega}(X)_{St}$ and $p, p' \in PCPattF$. I.e., *PCPattF* is the closure under conjunctions and disjunctions of $T_{\Omega}(X)_{St}$.

Positive Constructor Pattern Formulas

Positive constructor pattern formulas $PCPattF$ have the grammar:

$$u \;|\; p \vee p' \;|\; p \wedge p' \;|\; \bot$$

where $u \in T_{\Omega}(X)_{St}$ and $p, p' \in PCPattF$. I.e., $PCPattF$ is the closure under conjunctions and disjunctions of $T_{\Omega}(X)_{St}$. $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC and (recall from Lecture 18), $[\![p \vee p']\!] = [\![p]\!] \cup [\![p']\!]$,

Positive Constructor Pattern Formulas

Positive constructor pattern formulas $PCPattF$ have the grammar:

$$u \mid p \vee p' \mid p \wedge p' \mid \bot$$

where $u \in T_\Omega(X)_{St}$ and $p, p' \in PCPattF$. I.e., $PCPattF$ is the closure under conjunctions and disjunctions of $T_\Omega(X)_{St}$. $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC and (recall from Lecture 18), $[\![p \vee p']\!] = [\![p]\!] \cup [\![p']\!]$, and $[\![p \wedge p']\!] = [\![p]\!] \cap [\![p']\!]$.

Positive Constructor Pattern Formulas

Positive constructor pattern formulas *PCPattF* have the grammar:

$$u \;\mid\; p \vee p' \;\mid p \wedge p' \;\mid\; \bot$$

where $u \in T_{\Omega}(X)_{St}$ and $p, p' \in PCPattF$. I.e., *PCPattF* is the closure under conjunctions and disjunctions of $T_{\Omega}(X)_{St}$. $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC and (recall from Lecture 18), $[\![p \vee p']\!] = [\![p]\!] \cup [\![p']\!]$, and $[\![p \wedge p']\!] = [\![p]\!] \cap [\![p']\!]$. Of course, $[\![\bot]\!] = \emptyset$,

Positive Constructor Pattern Formulas

Positive constructor pattern formulas *PCPattF* have the grammar:

$$u \;\mid\; p \vee p' \;\mid p \wedge p' \;\mid\; \bot$$

where $u \in T_{\Omega}(X)_{St}$ and $p, p' \in \mathit{PCPattF}$. I.e., $\mathit{PCPattF}$ is the closure under conjunctions and disjunctions of $T_{\Omega}(X)_{St}$. $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC and (recall from Lecture 18), $[\![p \vee p']\!] = [\![p]\!] \cup [\![p']\!]$, and $[\![p \wedge p']\!] = [\![p]\!] \cap [\![p']\!]$. Of course, $[\![\bot]\!] = \emptyset$, and $[\![x : St]\!] = T_{\Omega/B,St}$.

Positive Constructor Pattern Formulas

Positive constructor pattern formulas $PCPattF$ have the grammar:

$$u \mid p \vee p' \mid p \wedge p' \mid \bot$$

where $u \in T_{\Omega}(X)_{St}$ and $p, p' \in PCPattF$. I.e., $PCPattF$ is the closure under conjunctions and disjunctions of $T_{\Omega}(X)_{St}$. $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC and (recall from Lecture 18), $[\![p \vee p']\!] = [\![p]\!] \cup [\![p']\!]$, and $[\![p \wedge p']\!] = [\![p]\!] \cap [\![p']\!]$. Of course, $[\![\bot]\!] = \emptyset$, and $[\![x : St]\!] = T_{\Omega/B,St}$. The proof of the following theorem can be found in Appendix 1:

Positive Constructor Pattern Formulas

Positive constructor pattern formulas $PCPattF$ have the grammar:

$$u \;\mid\; p \vee p' \;\mid p \wedge p' \;\mid\; \bot$$

where $u \in T_\Omega(X)_{St}$ and $p, p' \in PCPattF$. I.e., $PCPattF$ is the closure under conjunctions and disjunctions of $T_\Omega(X)_{St}$. $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC and (recall from Lecture 18), $[\![p \vee p']\!] = [\![p]\!] \cup [\![p']\!]$, and $[\![p \wedge p']\!] = [\![p]\!] \cap [\![p']\!]$. Of course, $[\![\bot]\!] = \emptyset$, and $[\![x\!:\!St]\!] = T_{\Omega/B,St}$. The proof of the following theorem can be found in Appendix 1:

DNF Theorem. Any $p \in PCPattF$ has a disjunctive normal form, $dnf(p)$, which is either $\bot$ or has the form $u_1 \vee \ldots \vee u_n$, with $u_i \in T_\Omega(X)_{St}$, $1 \leq i \leq n$, $n \geq 1$, and is such that $[\![p]\!] = [\![dnf(p)]\!]$.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas *NCPattF* have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

Negative Constructor Pattern Formulas

Negative constructor pattern formulas $NCPattF$ have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

where $u \in T_{\Omega}(X)_{St}$ and $n, n' \in PCPattF$.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas $NCPattF$ have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

where $u \in T_{\Omega}(X)_{St}$ and $n, n' \in PCPattF$. I.e., $NCPattF$ is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_{\Omega}(X)_{St}$.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas $NCPattF$ have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

where $u \in T_{\Omega}(X)_{St}$ and $n, n' \in PCPattF$. I.e., $NCPattF$ is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_{\Omega}(X)_{St}$. As before, $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas $NCPattF$ have the grammar:

$$\neg u \;\mid\; n \vee n' \;\mid n \wedge n' \;\mid\; \top$$

where $u \in T_{\Omega}(X)_{St}$ and $n, n' \in PCPattF$. I.e., $NCPattF$ is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_{\Omega}(X)_{St}$. As before, $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC. Recall form Lecture 18 that $[\![\neg u]\!] = T_{\Omega/B,St} \setminus [\![u]\!]$.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas $NCPattF$ have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

where $u \in T_{\Omega}(X)_{St}$ and $n, n' \in PCPattF$. I.e., $NCPattF$ is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_{\Omega}(X)_{St}$. As before, $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC. Recall form Lecture 18 that $[\![\neg u]\!] = T_{\Omega/B,St} \setminus [\![u]\!]$. Also, $[\![\top]\!] = T_{\Omega/B,St}$.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas $NCPattF$ have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

where $u \in T_\Omega(X)_{St}$ and $n, n' \in PCPattF$. I.e., $NCPattF$ is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_\Omega(X)_{St}$. As before, $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC. Recall form Lecture 18 that $[\![\neg u]\!] = T_{\Omega/B,St} \setminus [\![u]\!]$. Also, $[\![\top]\!] = T_{\Omega/B,St}$. The proof of the following theorem can be found in Appendix 1:

Negative Constructor Pattern Formulas

Negative constructor pattern formulas *NCPattF* have the grammar:

$$\neg u \mid n \vee n' \mid n \wedge n' \mid \top$$

where $u \in T_{\Omega}(X)_{St}$ and $n, n' \in PCPattF$. I.e., *NCPattF* is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_{\Omega}(X)_{St}$. As before, $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC. Recall form Lecture 18 that $[\![\neg u]\!] = T_{\Omega/B,St} \setminus [\![u]\!]$. Also, $[\![\top]\!] = T_{\Omega/B,St}$. The proof of the following theorem can be found in Appendix 1:

NCNF Theorem. Any $n \in NCPattF$ has a negative conjunctive normal form, $ncnf(n)$, with $ncnf(n)$ either $\top$ or of the form $\neg u_1 \wedge \ldots \wedge \neg u_n$, $u_i \in T_{\Omega}(X)_{St}$, $1 \leq i \leq n$, $n \geq 1$, and s.t. $[\![n]\!] = [\![ncnf(n)]\!]$.

Negative Constructor Pattern Formulas

Negative constructor pattern formulas *NCPattF* have the grammar:

$$\neg u \;\mid\; n \vee n' \;\mid\; n \wedge n' \;\mid\; \top$$

where $u \in T_\Omega(X)_{St}$ and $n, n' \in PCPattF$. I.e., *NCPattF* is the closure under conjunctions and disjunctions of negations $\neg u$ of patterns $u \in T_\Omega(X)_{St}$. As before, $\vee$ and $\wedge$ are assumed associative-commutative (AC), because $\cup$ and $\cap$ are AC. Recall form Lecture 18 that $[\![\neg u]\!] = T_{\Omega/B,St} \setminus [\![u]\!]$. Also, $[\![\top]\!] = T_{\Omega/B,St}$. The proof of the following theorem can be found in Appendix 1:

NCNF Theorem. Any $n \in NCPattF$ has a negative conjunctive normal form, $ncnf(n)$, with $ncnf(n)$ either $\top$ or of the form $\neg u_1 \wedge \ldots \wedge \neg u_n$, $u_i \in T_\Omega(X)_{St}$, $1 \leq i \leq n$, $n \geq 1$, and s.t. $[\![n]\!] = [\![ncnf(n)]\!]$. Note that $[\![\neg u_1 \wedge \ldots \wedge \neg u_n]\!] = T_{\Omega/B,St} \setminus [\![u_1 \vee \ldots \vee u_n]\!]$.

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v),

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$,

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$.

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command).

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$,

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_\Omega(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$,

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$,

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$, iff

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$, iff $\forall i$, $1 \leq i \leq n$, $\exists j$, $1 \leq j \leq n$, s.t., $u_i \sqsubseteq_B v_j$.

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_\Omega(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$, iff $\forall i$, $1 \leq i \leq n$, $\exists j$, $1 \leq j \leq n$, s.t., $u_i \sqsubseteq_B v_j$. Obviously, for B any combination of A and/or C and/or U axioms, the relation $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ is decidable.

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$, iff $\forall i$, $1 \leq i \leq n$, $\exists j$, $1 \leq j \leq n$, s.t., $u_i \sqsubseteq_B v_j$. Obviously, for B any combination of A and/or C and/or U axioms, the relation $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ is decidable. Likewise, $u_1 \vee \ldots \vee u_n$ is B-contained in $v_1 \vee \ldots \vee v_m$,

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$, iff $\forall i$, $1 \leq i \leq n$, $\exists j$, $1 \leq j \leq n$, s.t., $u_i \sqsubseteq_B v_j$. Obviously, for B any combination of A and/or C and/or U axioms, the relation $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ is decidable. Likewise, $u_1 \vee \ldots \vee u_n$ is B-contained in $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$,

Pattern Formula Containment and Subsumption

Definition. Given constructor patterns $u, v \in T_{\Omega}(X)_{St}$ and axioms B, we say that u is B-subsumed by v (or, equivalently, the v is B-more general than v), denoted $u \sqsubseteq_B v$, iff there exists a substitution α such that $u =_B v\alpha$. Note that for B any combination of A and/or C and/or U axioms, the relation $u \sqsubseteq_B v$ is decidable (e.g., by Maude's `match` command). Likewise, we say that u is B-contained in v, denoted $u \subseteq_B v$, iff $[\![u]\!] \subseteq [\![v]\!]$.

By definition, given positive pattern formulas $u_1 \vee \ldots \vee u_n$ and $v_1 \vee \ldots \vee v_m$, $n, m \geq 1$, (i) $\bot \sqsubseteq_B u_1 \vee \ldots \vee u_n$, and (ii) $u_1 \vee \ldots \vee u_n$ is B-subsumed by $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$, iff $\forall i$, $1 \leq i \leq n$, $\exists j$, $1 \leq j \leq n$, s.t., $u_i \sqsubseteq_B v_j$. Obviously, for B any combination of A and/or C and/or U axioms, the relation $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ is decidable. Likewise, $u_1 \vee \ldots \vee u_n$ is B-contained in $v_1 \vee \ldots \vee v_m$, denoted $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$, iff $[\![u_1 \vee \ldots \vee u_n]\!] \subseteq [\![v_1 \vee \ldots \vee v_m]\!]$.

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that:

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$.

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$.
(3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$.
(3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments,

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. (3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments, $u \subseteq_B v$ and $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$.

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. (3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments, $u \subseteq_B v$ and $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. However, by **Ex.24.2**-(3), this is not always a necessary condition.

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. (3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments, $u \subseteq_B v$ and $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. However, by **Ex.24.2**-(3), this is not always a necessary condition. The following lemma (the proof is left to the reader), can help us prove a containment $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$ by means of $\sqsubseteq_B$.

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. (3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments, $u \subseteq_B v$ and $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. However, by **Ex.24.2**-(3), this is not always a necessary condition. The following lemma (the proof is left to the reader), can help us prove a containment $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$ by means of $\sqsubseteq_B$.

Pattern Decompostion Lemma. For $u \in T_{\Omega/B}(X)_{St}$, $x\!:\!s \in vars(u)$, and $\{v_1, \ldots, v_m\}$ a generator set for sort s,

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. (3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments, $u \subseteq_B v$ and $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. However, by **Ex.24.2**-(3), this is not always a necessary condition. The following lemma (the proof is left to the reader), can help us prove a containment $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$ by means of $\sqsubseteq_B$.

Pattern Decompostion Lemma. For $u \in T_{\Omega/B}(X)_{St}$, $x\!:\!s \in vars(u)$, and $\{v_1, \ldots, v_m\}$ a generator set for sort s, we have the set equality:

Pattern Formula Containment and Subsumption (II)

Ex.24.2. Prove that: (1) $u \sqsubseteq_B v \Rightarrow u \subseteq_B v$, and (2) $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m \Rightarrow u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$.
(3) Give an example of constructor paterns u, v, w such that $u \not\sqsubseteq_B v \vee w$, but $u \subseteq_B v \vee w$.

The relations $u \sqsubseteq_B v$ and $u_1 \vee \ldots \vee u_n \sqsubseteq_B v_1 \vee \ldots \vee v_m$ give us a decidable sufficient condition to prove the corresponding B-containments, $u \subseteq_B v$ and $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$. However, by **Ex.24.2**-(3), this is not always a necessary condition. The following lemma (the proof is left to the reader), can help us prove a containment $u_1 \vee \ldots \vee u_n \subseteq_B v_1 \vee \ldots \vee v_m$ by means of $\sqsubseteq_B$.

Pattern Decompostion Lemma. For $u \in T_{\Omega/B}(X)_{St}$, $x{:}s \in vars(u)$, and $\{v_1, \ldots, v_m\}$ a generator set for sort s, we have the set equality: $[\![u]\!] = [\![u\{x{:}s \mapsto v_1\}]\!] \cup \ldots \cup [\![u\{x{:}s \mapsto v_m\}]\!]$. $\square$

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall $STACS$).

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall $STACS$). That is,

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall *STACS*). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$.

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall *STACS*). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition,

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall *STACS*). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$,

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall *STACS*). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \rightarrow^n_{R/B}[I]$,

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall *STACS*). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \rightarrow^n_{R/B}[I]$, and $\mathcal{R}^{\leq n}[I] =_{def} I \cup \mathcal{R}[I] \cup \ldots \cup \mathcal{R}^n[I]$, $n \in \mathbb{N}$.

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall $STACS$). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \rightarrow^n_{R/B}[I]$, and $\mathcal{R}^{\leq n}[I] =_{def} I \cup \mathcal{R}[I] \cup \ldots \cup \mathcal{R}^n[I]$, $n \in \mathbb{N}$.

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, a set $Q \subseteq T_{\Omega/B,St}$ is called $\mathcal{R}$-transition-closed iff $\mathcal{R}[Q] \subseteq Q$.

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall $STACS$). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \rightarrow^n_{R/B}[I]$, and $\mathcal{R}^{\leq n}[I] =_{def} I \cup \mathcal{R}[I] \cup \ldots \cup \mathcal{R}^n[I]$, $n \in \mathbb{N}$.

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, a set $Q \subseteq T_{\Omega/B,St}$ is called $\mathcal{R}$-transition-closed iff $\mathcal{R}[Q] \subseteq Q$. Also, $Q \subseteq T_{\Omega/B,St}$ is called an inductive invariant from initial states $I \subseteq T_{\Omega/B,St}$ iff

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\to^*_{R/B}[I]$ (recall $STACS$). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \to^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \to_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \to^n_{R/B}[I]$, and $\mathcal{R}^{\leq n}[I] =_{def} I \cup \mathcal{R}[I] \cup \ldots \cup \mathcal{R}^n[I]$, $n \in \mathbb{N}$.

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, a set $Q \subseteq T_{\Omega/B,St}$ is called $\mathcal{R}$-transition-closed iff $\mathcal{R}[Q] \subseteq Q$. Also, $Q \subseteq T_{\Omega/B,St}$ is called an inductive invariant from initial states $I \subseteq T_{\Omega/B,St}$ iff (i) Q is an invariant from I, i.e., $\mathcal{R}^*[I] \subseteq Q$, and

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall *STACS*). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \rightarrow^n_{R/B}[I]$, and $\mathcal{R}^{\leq n}[I] =_{def} I \cup \mathcal{R}[I] \cup \ldots \cup \mathcal{R}^n[I]$, $n \in \mathbb{N}$.

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, a set $Q \subseteq T_{\Omega/B,St}$ is called $\mathcal{R}$-transition-closed iff $\mathcal{R}[Q] \subseteq Q$. Also, $Q \subseteq T_{\Omega/B,St}$ is called an inductive invariant from initial states $I \subseteq T_{\Omega/B,St}$ iff (i) Q is an invariant from I, i.e., $\mathcal{R}^*[I] \subseteq Q$, and (ii) Q is $\mathcal{R}$-transition-closed.

Reachable, Transition-Closed, and Inductive Invariants

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, the set $\mathcal{R}^*[I]$ of $\mathcal{R}$-reachable states from a set $I \subseteq T_{\Omega/B,St}$ of initial states is, by definition, the set $\rightarrow^*_{R/B}[I]$ (recall $STACS$). That is, $\mathcal{R}^*[I] = \{[v] \in T_{\Omega/B,St} \mid \exists [u] \in I \;\; s.t. \;\; [u] \rightarrow^*_{R/B} [v]\}$. Likewise, by definition, $\mathcal{R}[I] =_{def} \rightarrow_{R/B}[I]$, $\mathcal{R}^n[I] =_{def} \rightarrow^n_{R/B}[I]$, and $\mathcal{R}^{\leq n}[I] =_{def} I \cup \mathcal{R}[I] \cup \ldots \cup \mathcal{R}^n[I]$, $n \in \mathbb{N}$.

Definition. For $\mathcal{R} = (\Omega, B, R)$ topmost with state sort St, a set $Q \subseteq T_{\Omega/B,St}$ is called $\mathcal{R}$-transition-closed iff $\mathcal{R}[Q] \subseteq Q$. Also, $Q \subseteq T_{\Omega/B,St}$ is called an inductive invariant from initial states $I \subseteq T_{\Omega/B,St}$ iff (i) Q is an invariant from I, i.e., $\mathcal{R}^*[I] \subseteq Q$, and (ii) Q is $\mathcal{R}$-transition-closed.

Ex.24.1: Prove that: (1) Q is $\mathcal{R}$-transition-closed iff $\mathcal{R}^*[Q] = Q$, and (2) the smallest invariant from a set of initial states $I \subseteq T_{\Omega/B,St}$, namely, $\mathcal{R}^*[I]$, is inductive.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures).

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w. For each depth $d \in \mathbb{N}$ the algorithm iteratively computes positive pattern formulas P_d and F_d, with $F_d \sqsubseteq_B P_d$ and such that

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w. For each depth $d \in \mathbb{N}$ the algorithm iteratively computes positive pattern formulas P_d and F_d, with $F_d \sqsubseteq_B P_d$ and such that $\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!] = [\![P_d]\!]$.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w. For each depth $d \in \mathbb{N}$ the algorithm iteratively computes positive pattern formulas P_d and F_d, with $F_d \sqsubseteq_B P_d$ and such that $\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!] = [\![P_d]\!]$. The algorithm (searching for one solution) terminates if a smallest d is reached s.t. either:

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w. For each depth $d \in \mathbb{N}$ the algorithm iteratively computes positive pattern formulas P_d and F_d, with $F_d \sqsubseteq_B P_d$ and such that $\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!] = [\![P_d]\!]$. The algorithm (searching for one solution) terminates if a smallest d is reached s.t. either: (i) $F_d \wedge w \neq \bot$ (a decidable property: see Appendix 1), i.e., a solution is found, or

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w. For each depth $d \in \mathbb{N}$ the algorithm iteratively computes positive pattern formulas P_d and F_d, with $F_d \sqsubseteq_B P_d$ and such that $\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!] = [\![P_d]\!]$. The algorithm (searching for one solution) terminates if a smallest d is reached s.t. either: (i) $F_d \wedge w \neq \bot$ (a decidable property: see Appendix 1), i.e., a solution is found, or (ii) $F_d = \bot$, i.e., no solution exists, proving that $[\![\neg w]\!]$ is an invariant from $[\![u_1 \vee \ldots \vee u_n]\!]$.

The Folding Narrowing Search Algorithm

Folding narrowing search is a symbolic model checking algorithm to verify invariants of infinite-state systems. It applies to topmost rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$ with $E \cup B$ enjoying the finite variant property (FVP) (more on this in future lectures). Here we focus on $\mathcal{R} = (\Omega, B, R)$ topmost. The search algorithm is the same, just generalizing $\rightsquigarrow_{R/B}$ to $\rightsquigarrow_{R/E \cup B}$.

The initial state is a possitive pattern formula of the form, $u_1 \vee \ldots \vee u_n$. The goal state is a pattern w. For each depth $d \in \mathbb{N}$ the algorithm iteratively computes positive pattern formulas P_d and F_d, with $F_d \sqsubseteq_B P_d$ and such that $\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!] = [\![P_d]\!]$. The algorithm (searching for one solution) terminates if a smallest d is reached s.t. either: (i) $F_d \wedge w \neq \bot$ (a decidable property: see Appendix 1), i.e., a solution is found, or (ii) $F_d = \bot$, i.e., no solution exists, proving that $[\![\neg w]\!]$ is an invariant from $[\![u_1 \vee \ldots \vee u_n]\!]$. Otherwise, the search loops forever.

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d.

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \leadsto_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \rightsquigarrow_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

where the notation $\bigvee$ generalizes the pattern disjunction operation $\vee$ to any finite set of patterns,

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \rightsquigarrow_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

where the notation $\bigvee$ generalizes the pattern disjunction operation $\vee$ to any finite set of patterns, e.g., $\bigvee\{v_1, \ldots, v_m\} = v_1 \vee \ldots \vee v_m$.

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \leadsto_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

where the notation $\bigvee$ generalizes the pattern disjunction operation $\vee$ to any finite set of patterns, e.g., $\bigvee\{v_1, \ldots, v_m\} = v_1 \vee \ldots \vee v_m$. That is, F_{d+1} excludes all w such that $v_i \leadsto_{R/B} w$ and $w \sqsubseteq_B P_d$,

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \rightsquigarrow_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

where the notation $\bigvee$ generalizes the pattern disjunction operation $\vee$ to any finite set of patterns, e.g., $\bigvee\{v_1, \ldots, v_m\} = v_1 \vee \ldots \vee v_m$. That is, F_{d+1} excludes all w such that $v_i \rightsquigarrow_{R/B} w$ and $w \sqsubseteq_B P_d$, i.e., those w that "fold" into P_d.

The Folding Narrowing Search Algorithm (II)

The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \rightsquigarrow_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

where the notation $\bigvee$ generalizes the pattern disjunction operation $\vee$ to any finite set of patterns, e.g., $\bigvee\{v_1, \ldots, v_m\} = v_1 \vee \ldots \vee v_m$. That is, F_{d+1} excludes all w such that $v_i \rightsquigarrow_{R/B} w$ and $w \sqsubseteq_B P_d$, i.e., those w that "fold" into P_d. Call F_d the frontier of P_d, $d \in \mathbb{N}$.

The Folding Narrowing Search Algorithm (II)

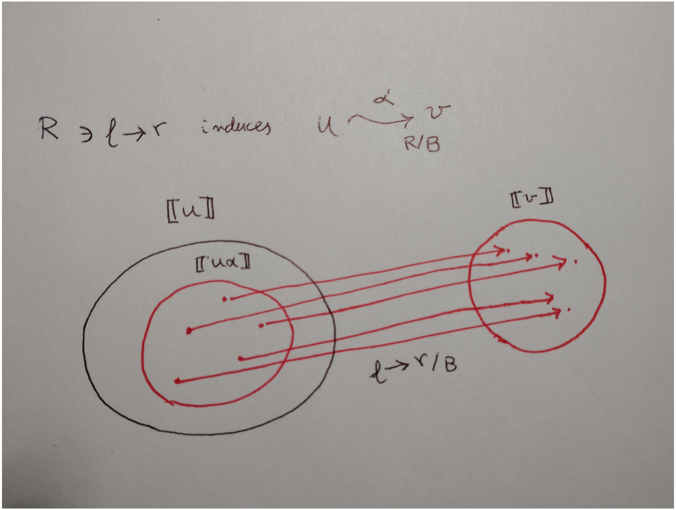
The positive pattern formulas $P_d(u_1 \vee \ldots \vee u_n)$ and $F_d(u_1 \vee \ldots \vee u_n)$ associated to a set of initial states $[\![u_1 \vee \ldots \vee u_n]\!]$ are abbreviated to P_d and F_d. They are computed inductively for increasing depth $d \in \mathbb{N}$ as follows:

- $P_0 = F_0 = u_1 \vee \ldots \vee u_n$.
- $P_{d+1} = P_d \vee F_{d+1}$, where for $F_d = v_1 \vee \ldots \vee v_m$,

$$F_{d+1} = \bigvee \{w \mid \exists i,\ 1 \leq i \leq m,\ s.t., v_i \leadsto_{R/B} w \wedge w \not\sqsubseteq_B P_d\}.$$

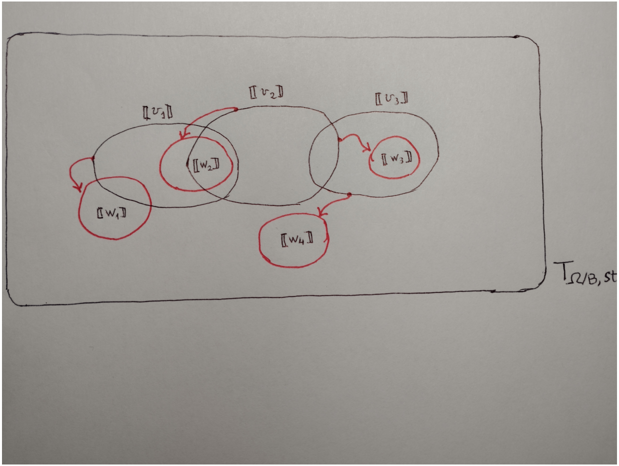
where the notation $\bigvee$ generalizes the pattern disjunction operation $\vee$ to any finite set of patterns, e.g., $\bigvee\{v_1, \ldots, v_m\} = v_1 \vee \ldots \vee v_m$. That is, F_{d+1} excludes all w such that $v_i \leadsto_{R/B} w$ and $w \sqsubseteq_B P_d$, i.e., those w that "fold" into P_d. Call F_d the frontier of P_d, $d \in \mathbb{N}$. The algorithm terminates for the smallest d (if any) s.t. either $F_d \wedge v \neq \bot$ for goal state v, or $F_d = \bot$.

The Set-Theoretic Meaning of Narrowing



The Set-Theoretic Meaning of Folding Narrowing

For $F_d = v_1 \vee v_2 \vee v_3$, then $F_{d+1} = w_1 \vee w_2 \vee w_4$. w_3 folded into v_3.



Completeness of Folding Narrowing

Completeness Theorem of Folding Narrowing. Let (Ω, B, R) be a topmost rewrite theory with state sort St, and $u_1 \vee \ldots \vee u_n$ an initital state. For each depth $d \in \mathbb{N}$, $[\![P_d]\!] = \mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!]$.

If it exists, let d be the smallest depth such that $F_{d+1} = \bot$. Then, $P_{d+1} = P_d \vee F_{d+1} = P_d \vee \bot$, which implies $[\![P_d]\!] = [\![P_{d+1}]\!]$. I.e., $\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!] = [\![P_d]\!] = [\![P_{d+1}]\!] = \mathcal{R}^{\leq d+1}[\![u_1 \vee \ldots \vee u_n]\!] = \mathcal{R}[\mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!]] \cup \mathcal{R}^{\leq d}[\![u_1 \vee \ldots \vee u_n]\!]$, so that $[\![P_d]\!]$ is transition-closed. Therefore, by **Ex.24.1** we have $[\![P_d]\!] = \mathcal{R}^*[\![P_d]\!]$. But then $[\![P_d]\!] = \mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!]$ follows from the inclusions:

$$\mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!] \subseteq \mathcal{R}^*[\![P_d]\!] = [\![P_d]\!] \subseteq \mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!].$$

That is, we get a finite, symbolic descrition of all reachable states $\mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!]$ as the pattern disjunction P_d.

Four Methods to Symbolically Verify Invariants

For $\mathcal{R} = (\Omega, B, R)$ a topmost rewrite theory with state sort St, $u_1 \vee \ldots \vee u_n$ an inititial state, and $Q \subseteq T_{\Omega/B,St}$, the following four methods can verify (†) $\mathbb{C}_{\mathcal{R}}, [\![u_1 \vee \ldots \vee u_n]\!] \models_{S4} \Box Q$.

A. If Q is specifiable as $Q = [\![n]\!]$ for n a negative pattern formula different from $\top$ (if $n = \top$, (†) holds trivially). W.L.O.G. we may assume $n = ncnf(n) = \neg v_1 \wedge \ldots \wedge \neg v_m$.

Method 1. (†) holds if $\mathbb{C}_{\mathcal{R}}, [\![u_1 \vee \ldots \vee u_n]\!] \not\models_{S4} \Diamond[\![v_1 \vee \ldots \vee v_m]\!]$. A sufficient condition to automatically verify (†) is that the m commands `{fold} vu-narrow` $u_1 \vee \ldots \vee u_n$ `=>*` v_j, $1 \leq j \leq m$ return: `No solution`.

If this succeeds, Maude can retun the positive pattern disjunction P_d such that $[\![P_d]\!] = \mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!]$, which enables **Method 2**.

Four Methods to Symbolically Verify Invariants (II)

Method 2. If we have found $P_d = w_1 \vee \ldots \vee w_k$ s.t. $[\![P_d]\!] = \mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!]$, then (†) holds for any Q of the form, $Q = [\![\neg v_1 \wedge \ldots \wedge \neg v_m]\!]$ iff $\forall 1 \leq i \leq k, \forall 1 \leq j \leq m, w_i \wedge v_j = \bot$, i.e., (see Appendix 1), iff $Unif_B(w_i = v_j) = \emptyset$ for all i, j (we assume $vars(w_i) = vars(v_j)$). Note that no search is needed!

B. If Q is specifiable as $Q = [\![p]\!]$ for p a positive pattern formula different from $\bot$ (if $p = \bot$, (†) cannot hold). W.L.O.G. we may assume $p = dnf(p) = v_1 \vee \ldots \vee v_m$.

Method 3. If we have found $P_d = w_1 \vee \ldots \vee w_k$ s.t. $[\![P_d]\!] = \mathcal{R}^*[\![u_1 \vee \ldots \vee u_n]\!]$, then (†) holds for any Q of the form, $Q = [\![v_1 \vee \ldots \vee v_m]\!]$ iff $w_1 \vee \ldots \vee w_k \subseteq_B v_1 \vee \ldots \vee v_m$. A decidable sufficient condition is $w_1 \vee \ldots \vee w_k \sqsubseteq_B v_1 \vee \ldots \vee v_m$.

Four Methods to Symbolically Verify Invariants (III)

Method 4. (†) holds for $Q = [\![v_1 \vee \ldots \vee v_m]\!]$ if: (1) Q is transitition-closed; this holds iff a @fold vu-narrow $v_1 \vee \ldots \vee v_m$ =>1 $ command, where $ is a fresh (and therefore unreachable) constant added to $\mathcal{R}$, generates an $F_1(v_1 \vee \ldots \vee v_m)$ s.t. either $F_1(v_1 \vee \ldots \vee v_m) = \bot$, or $F_1(v_1 \vee \ldots \vee v_m) \subset_B v_1 \vee \ldots \vee v_m$. (2) $u_1 \vee \ldots \vee u_n \subset_B v_1 \vee \ldots \vee v_m$. A decidable sufficient condition is $u_1 \vee \ldots \vee u_n \sqsubset_B v_1 \vee \ldots \vee v_m$.