Representations

Michal Štolba
stolba@agents.fel.cvut.cz

Department of Computer Science, CTU

PUI (Planning in Artificial Intelligence)

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$
  - P finite set of propositional (true/false) variables
  - O finite set of operators
- Set representation
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  pre: $\{p \in P | p = \text{true}\}$
  add: $\{p \in P | p \leftarrow \text{true}\}$
  del: $\{p \in P | p \leftarrow \text{false}\}$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  pre: $\{p \in P | p = \text{true}\}$
  add: $\{p \in P | p \leftarrow \text{true}\}$
  del: $\{p \in P | p \leftarrow \text{false}\}$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  pre: $\{p \in P | p = \text{true}\}$
  add: $\{p \in P | p \leftarrow \text{true}\}$
  del: $\{p \in P | p \leftarrow \text{false}\}$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\text{pre: } \{p \in P | p = \text{true}\}$$
  $$\text{add: } \{p \in P | p \leftarrow \text{true}\}$$
  $$\text{del: } \{p \in P | p \leftarrow \text{false}\}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\text{pre: } \{p \in P | p = \text{true}\}$$
  $$\text{add: } \{p \in P | p \leftarrow \text{true}\}$$
  $$\text{del: } \{p \in P | p \leftarrow \text{false}\}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre: } & \{p \in P | p = \text{true}\} \\ \text{add: } & \{p \in P | p \leftarrow \text{true}\} \\ \text{del: } & \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre:} &\ \{p \in P | p = \text{true}\} \\ \text{add:} &\ \{p \in P | p \leftarrow \text{true}\} \\ \text{del:} &\ \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = $ true, other false)

  G goal state ($p \in P$ s.t. $p = $ true; $p' \in P$ s.t. $p = $ false)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

$$\begin{aligned} \text{pre: } & \{p \in P | p = \text{true}\} \\ \text{add: } & \{p \in P | p \leftarrow \text{true}\} \\ \text{del: } & \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre:} &\ \{p \in P | p = \text{true}\} \\ \text{add:} &\ \{p \in P | p \leftarrow \text{true}\} \\ \text{del:} &\ \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  - **P** finite set of propositional (true/false) variables
  - **O** finite set of operators:

$$\begin{aligned}\text{pre: } & \{p \in P | p = \text{true}\} \\ \text{add: } & \{p \in P | p \leftarrow \text{true}\} \\ \text{del: } & \{p \in P | p \leftarrow \text{false}\}\end{aligned}$$

  - **I** initial state ($p \in P$ s.t. $p = $ true, other false)
  - **G** goal state ($p \in P$ s.t. $p = $ true; $p' \in P$ s.t. $p = $ false)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

$$\begin{aligned} \text{pre: } & \{p \in P | p = \text{true}\} \\ \text{add: } & \{p \in P | p \leftarrow \text{true}\} \\ \text{del: } & \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = $ true, other false)

  G goal state ($p \in P$ s.t. $p = $ true; $p' \in P$ s.t. $p = $ false)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$
  - P finite set of propositional (true/false) variables
  - O finite set of operators:

$$\begin{aligned} \text{pre:} &\ \{p \in P | p = \text{true}\} \\ \text{add:} &\ \{p \in P | p \leftarrow \text{true}\} \\ \text{del:} &\ \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  - I initial state ($p \in P$ s.t. $p = \text{true}$, other false)
  - G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)
- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre:} & \ \{p \in P | p = \text{true}\} \\ \text{add:} & \ \{p \in P | p \leftarrow \text{true}\} \\ \text{del:} & \ \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = $ true, other false)

  G goal state ($p \in P$ s.t. $p = $ true; $p' \in P$ s.t. $p = $ false)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre:} \quad & \{p \in P | p = \text{true}\} \\ \text{add:} \quad & \{p \in P | p \leftarrow \text{true}\} \\ \text{del:} \quad & \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre: } & \{p \in P | p = \text{true}\} \\ \text{add: } & \{p \in P | p \leftarrow \text{true}\} \\ \text{del: } & \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

(STanford Research Institute Problem Solver)

- 1966-1972 – Shakey the Robot
- $\langle P, O, I, G \rangle$

  P finite set of propositional (true/false) variables

  O finite set of operators:

  $$\begin{aligned} \text{pre:} & \ \{p \in P | p = \text{true}\} \\ \text{add:} & \ \{p \in P | p \leftarrow \text{true}\} \\ \text{del:} & \ \{p \in P | p \leftarrow \text{false}\} \end{aligned}$$

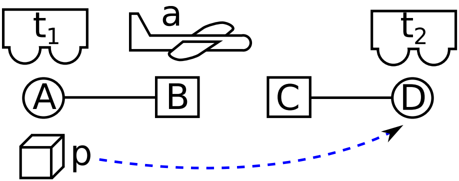
  I initial state ($p \in P$ s.t. $p = \text{true}$, other false)

  G goal state ($p \in P$ s.t. $p = \text{true}$; $p' \in P$ s.t. $p = \text{false}$)

- Set representation
  - true/false determined by the set membership
- Plan existence PSPACE-Complete

STRIPS

Example



STRIPS

Example

- *P* propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

STRIPS

Example

- *P* propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

STRIPS

Example

- P propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

STRIPS

Example

- *P* propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

STRIPS

Example

- P propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

STRIPS

Example

- P propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

STRIPS
Example

- *O* operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}

  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS

Example

- O operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}

  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS

Example

- *O* operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}
  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS
Example

- O operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}

  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS

Example

- O operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}
  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS

Example

- O operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}
  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS

Example

- O operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}
  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS

Example

- *O* operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}

  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

STRIPS
Example

- *O* operators:
  - load-p-a-B

    pre: {plane-at-B, package-at-B}
    add: {package-in-a}
    del: {package-at-B}
  - move-t-A-B

    pre: {truck-at-A}
    add: {truck-at-B}
    del: {truck-at-A}

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

V finite set of state variables v with associated finite domain D_v

- If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
- Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times \ldots \times v_k \longmapsto D_{v_1} \times \ldots \times D_{v_k}$
- State is a partial state defined on all variables in V

i The initial state

g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables v with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times ... \times v_k \longmapsto D_{v_1} \times ... \times D_{v_k}$
  - State is a partial state defined on all variables in V

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables v with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times \ldots \times v_k \longmapsto D_{v_1} \times \ldots \times D_{v_k}$
  - State is a partial state defined on all variables in V

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables v with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times ... \times v_k \longmapsto D_{v_1} \times ... \times D_{v_k}$
  - State is a partial state defined on all variables in V

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables v with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times ... \times v_k \longmapsto D_{v_1} \times ... \times D_{v_k}$
  - State is a partial state defined on all variables in V

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables v with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times ... \times v_k \longmapsto D_{v_1} \times ... \times D_{v_k}$
  - State is a partial state defined on all variables in V

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables v with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state s defined on $V' \subseteq V$ variables is a function $s : v_1 \times ... \times v_k \longmapsto D_{v_1} \times ... \times D_{v_k}$
  - State is a partial state defined on all variables in V

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  V finite set of state variables *v* with associated finite domain D_v

  - If $D_v = \{\text{true}, \text{false}\}$ equivalent to STRIPS
  - Partial state *s* defined on $V' \subseteq V$ variables is a function $s : v_1 \times ... \times v_k \longmapsto D_{v_1} \times ... \times D_{v_k}$
  - State is a partial state defined on all variables in *V*

  i The initial state

  g Partial state defining the goal

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state
  eff: partial state

- Application of o in state s resulting in s'
  - Values of variables defined in pre(o) must be equal to their values in s
  - Values in of variables in s' are
    set to values in eff(o) if defined
    set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state
  eff: partial state

- Application of o in state s resulting in s'
  - Values of variables defined in pre(o) must be equal to their values in s
  - Values in of variables in s' are
    set to values in eff(o) if defined
    set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state

  eff: partial state

- Application of o in state s resulting in s'
  - Values of variables defined in pre(o) must be equal to their values in s
  - Values in of variables in s' are
    set to values in eff(o) if defined
    set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state
  eff: partial state

- Application of o in state s resulting in s'
  - Values of variables defined in pre(o) must be equal to their values in s
  - Values in of variables in s' are
    set to values in eff(o) if defined
    set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state

  eff: partial state

- Application of o in state s resulting in s'
  - Values of variables defined in pre(o) must be equal to their values in s
  - Values in of variables in s' are
    set to values in eff(o) if defined
    set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state
  eff: partial state

- Application of o in state s resulting in s'
  - Values of variables defined in pre(o) must be equal to their values in s
  - Values in of variables in s' are
    set to values in eff(o) if defined
    set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state
  eff: partial state

  - Application of o in state s resulting in s'
    - Values of variables defined in pre(o) must be equal to their values in s
    - Values in of variables in s' are
      set to values in eff(o) if defined
      set to values in s else

MPT/FDR

Multi-valued Planning Task / Finite Domain Representation (or SAS+)

- $\langle V, O, i, g \rangle$

  O finite set of operators $o = \langle \text{pre}(o), \text{eff}(o) \rangle$ where

  pre: partial state
  eff: partial state

  - Application of o in state s resulting in s'
    - Values of variables defined in pre(o) must be equal to their values in s
    - Values in of variables in s' are
      set to values in eff(o) if defined
      set to values in s else

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B: pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B: pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B: pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B: pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B pre: plane-at=B, package-at=B
eff: package-at=a
move-t-A-B pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B: pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B: pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B: pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B: pre: truck-at=A
eff: truck-at=B

MPT/FDR

Example

V variables and their domains:

- truck-at $\in \{A, B\}$
- plane-at $\in \{B, C\}$
- package-at $\in \{A, B, C, t, a\}$
- $2 \times 2 \times 5 = 20$ states

O operators:

load-p-a-B: pre: plane-at=B, package-at=B
eff: package-at=a

move-t-A-B: pre: truck-at=A
eff: truck-at=B

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

Domain file definition of types, predicates, operators
Problem file definition of objects, initial state and goal

- Lisp-like syntax
- Prefix notation (+1 2)
- A lot of brackets
- Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

Domain file definition of types, predicates, operators

Problem file definition of objects, initial state and goal

  - Lisp-like syntax
  - Prefix notation (+1 2)
  - A lot of brackets
  - Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

  Domain file definition of types, predicates, operators
  Problem file definition of objects, initial state and goal

    - Lisp-like syntax
    - Prefix notation $(+ 1 2)$
    - A lot of brackets
    - Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

  Domain file definition of types, predicates, operators
  Problem file definition of objects, initial state and goal

  - Lisp-like syntax
  - Prefix notation (+1 2)
  - A lot of brackets
  - Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

  Domain file definition of types, predicates, operators

  Problem file definition of objects, initial state and goal

    - Lisp-like syntax
    - Prefix notation $(+1\,2)$
    - A lot of brackets
    - Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

Domain file definition of types, predicates, operators
Problem file definition of objects, initial state and goal

  - Lisp-like syntax
  - Prefix notation $(+ 1\, 2)$
  - A lot of brackets
  - Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

  Domain file definition of types, predicates, operators
  Problem file definition of objects, initial state and goal

  - Lisp-like syntax
  - Prefix notation $(+1\,2)$
  - A lot of brackets
  - Several versions (1.2, 2.1, 3.1)

PDDL

Planning Domain Definition Language

- General language (predicate logic) to describe planning problems

Domain file definition of types, predicates, operators
Problem file definition of objects, initial state and goal

  - Lisp-like syntax
  - Prefix notation $(+1\,2)$
  - A lot of brackets
  - Several versions (1.2, 2.1, 3.1)

PDDL

Grounding

- Predicate logic
  - → STRIPS (propositional logic)
    - All possible instantiations of predicates → propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - → FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis – variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - → STRIPS (propositional logic)
    - All possible instantiations of predicates → propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - → FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis → variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis – variables and domains

PDDL

Grounding

- Predicate logic
  - → STRIPS (propositional logic)
    - All possible instantiations of predicates → propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - → FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis → variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL

Grounding

- Predicate logic
  - $\rightarrow$ STRIPS (propositional logic)
    - All possible instantiations of predicates $\rightarrow$ propositions
    - All possible instantiations of actions
    - Is it all necessary?
    - No! Reachability analysis can help
  - $\rightarrow$ FDR (finite domain representation)
    - Grounding as in STRIPS
    - Invariant analysis - variables and domains

PDDL
Resources

- Online editor:
  http://editor.planning.domains
- Resources:
  http://ipc.informatik.uni-freiburg.de/PddlResources
  http://users.cecs.anu.edu.au/~patrik/pddlman/writing.html
  http://www.cs.toronto.edu/~sheila/2542/f10/A1/introtopddl2.pdf
  http://en.wikipedia.org/wiki/Planning_Domain_Definition_Language

PDDL

Resources

- Online editor:
  http://editor.planning.domains
- Resources:
  http://ipc.informatik.uni-freiburg.de/PddlResources
  http://users.cecs.anu.edu.au/~patrik/pddlman/writing.html
  http://www.cs.toronto.edu/~sheila/2542/f10/A1/introtopddl2.pdf
  http://en.wikipedia.org/wiki/Planning_Domain_Definition_Language