Automatic Polytime Reductions of NP Problems into a Fragment of STRIPS

Aldo Porco    Alejandro Machado    Blai Bonet

Universidad Simón Bolívar, Caracas, Venezuela

ICAPS 2011 – Freiburg, June 2011
Motivation

- for using planners, one needs to come up with sound PDDLs
- even if you know well the problem, it may not be easy to translate it into PDDL
- it would be nice to automatically translate problems described in high-level declarative language into PDDL
Our Contribution

- tool that automatically translates NP problems into PDDL
- problems specified using logic in declarative manner
- translation runs in polytime
- existence of plans for generated PDDLS can be decided in NP
- tool fully characterized by its formal properties
This Talk

- Descriptive Complexity Theory
- Tool
- Translations
- Experiments
- Discussion
Branch of Complexity Theory that uses logic instead of TMs to characterize complexity classes

In Descriptive Complexity Theory (DCT):

- problem corresponds to collection of finite structures
- collection is the set of finite models for a logic formula
- complexity class (class of problems) corresponds to a fragment of logic
For example, NP equals all problems definable in the existential fragment of second-order logic (SO∃)

**Main results of DCT:**

- P equals SO-Horn
- NP equals SO∃ and coNP equals SO∀
- Polynomial-time hierarchy (PH) equals SO
- PSPACE equals SO + Transitive Closure (SO+TC)

E.g., PH = PSPACE iff TC does not add expressivity to SO
Example: SAT

CNF $\varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1)$

clause 0

clause 1

clause 2
Example: SAT

CNF $\varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1)$

clause 0 \hspace{1cm} clause 1 \hspace{1cm} clause 2

Encoded with relations $P(x, y)$ and $N(x, y)$ interpreted as:

- $P(x, y)$ iff variable $x$ appears positive in clause $y$
- $N(x, y)$ iff variable $x$ appears negative in clause $y$
Example: SAT

CNF $\varphi = \left( x_0 \lor \neg x_1 \lor x_2 \right) \land \left( \neg x_0 \lor \neg x_2 \right) \land \left( \neg x_0 \lor x_1 \right)$

clause 0

clause 1

clause 2

E.g., $\varphi$ encoded with structure $\mathcal{A} = \langle |\mathcal{A}|, P^\mathcal{A}, N^\mathcal{A} \rangle$ where

- universe is $|\mathcal{A}| = \{0, 1, 2\}$
- interpretation of $P$ is $P^\mathcal{A} = \{(0, 0), (2, 0), (1, 2)\}$
- interpretation of $N$ is $N^\mathcal{A} = \{(1, 0), (0, 1), (2, 1), (0, 2)\}$
Example: SAT

\[ \varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1) \]

Clause 0

Clause 1

Clause 2

Model: \( \{ x_0, x_1, \neg x_2 \} \) encoded with unary relation \( T(x) \) such that

- \( T(x_0) \)
- \( T(x_1) \)
- \( \neg T(x_2) \)

I.e., \( T \) has interpretation \( \{0, 1\} \)
Example: SAT

CNF $\varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1)$

- clause 0
- clause 1
- clause 2

Extended structure $\langle |A|, P^A, N^A, T \rangle$ is model of

$$(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]$$

iff $T$ encodes a model of $\varphi$
Example: SAT

CNF $\varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1)$

Extended structure $\langle |A|, P^A, N^A, T \rangle$ is model of

$$(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]$$

iff $T$ encodes a model of $\varphi$

Hence, $\varphi$ is SATISFIABLE iff $A$ is model of

$$\Phi = (\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]$$
Example: SAT

\[ \text{CNF } \varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1) \]

\text{clause 0} \quad \text{clause 1} \quad \text{clause 2}

Indeed, SAT = MOD[\Phi]
Example: SAT

CNF $\varphi = (x_0 \lor \neg x_1 \lor x_2) \land (\neg x_0 \lor \neg x_2) \land (\neg x_0 \lor x_1)$

Indeed, $\text{SAT} = \text{MOD}[\Phi]$

Meaning:

- for every satisfiable formula $\varphi$, its encoding $A_\varphi \in \text{MOD}[\Phi]$
- for every $A \in \text{MOD}[\Phi]$, $A$ encodes a satisfiable formula $\varphi_A$
Example: 3-Colorability

Signature

- $E(x, y)$: undirected edge linking nodes $x$ and $y$ in the graph

Formula

Every node must be colored with single color; if two nodes are connected, their colors must be different

$$(\exists R^1, G^1, B^1)(\forall x, y)[$$

$$(R(x) \lor G(x) \lor B(x)) \land$$

$$R(x) \rightarrow \neg(G(x) \lor B(x)) \land$$

$$G(x) \rightarrow \neg(R(x) \lor B(x)) \land$$

$$B(x) \rightarrow \neg(R(x) \lor G(x)) \land$$

$$E(x, y) \rightarrow \neg[(R(x) \land R(y)) \lor (G(x) \land G(y)) \lor (B(x) \land B(y))]$$]
Example: Directed Hamiltonian Path

Signature

- $E(x, y)$: directed edge linking nodes $x$ and $y$ in the graph

Formula

A DHP is a sequence vertices such that there is directed edge from $a_i$ to $a_{i+1}$ for every vertex $a_i < max$. It can be seen as injective function $F : [0...n] \rightarrow |\mathcal{A}|$

$$(\exists F \in \text{Inj})(\forall x)[x < \text{max} \rightarrow (\exists x'yz)(E(y, z) \land F(x, y) \land \text{SUC}(x, x') \land F(x', z))]$$
The Tool
Input:
- signature $\sigma$ that contains relational symbols
- SO∃ formula $\Phi$ that encodes NP problem
- finite structure $A$ that encodes instance

Output:
- PDDLs for a fragment of STRIPS that is decidable in NP

Guarantees:
- runs in polytime for fixed $\Phi$
- output is no harder than input (complexity-wise)
Related Work

- DATALOG-like specification of NP problems into SAT (Cadoli & Schaerf, 2005)
  - we are targeting STRIPS
  - we would like to go beyond NP

- Framework for describing problems based on the Model Extension (MX) (Mitchell & Ternovska, 2005)
  - translated problems are solvable using planning technology
Translation
Two Steps

Translation divided in two steps:

- generation of PDDL domain
- generation of PDDL problem instance

Can be thought as two functions:

\[ \mathcal{D} : \text{Signatures} \times \text{SO}\exists \rightarrow \text{PDDL Domains} \]

\[ \mathcal{I} : \text{Signatures} \times \text{SO}\exists \times \text{STRUC} \rightarrow \text{PDDL Instances} \]
Different Translations

Translations that aim different planners:

- for sequential planners
- for parallel planners
- for optimal sequential planners
Domain $\mathcal{D}(\sigma, \Phi)$:

- $\Phi$ assumed to have negations only at literal level
- two predicates for each relational symbol $P$
  - $P(?x)$
  - $\text{not-}P(?x)$
- operators that add positive fluents for quantified relations
- for each FO subformula $\theta$ of $\Phi$, except literals, there are
  - fluent that denote the validity of $\theta$ wrt extended $A$
  - operators that add the fluent
Translation Used in Experiments

Instance $\mathcal{I}(\sigma, \Phi, \mathcal{A})$:

- objects for each element in universe $|\mathcal{A}|$
- initial situation:
  - fluents for interpretations in $\mathcal{A}$
  - all ‘not–’ fluents for quantified relations (SO)
Example: SAT – Fluents

\[ (\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))] \]

(T ?x) (not-T ?x)
Example: SAT – Fluents

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]]

- (T ?x) (not-T ?x)
- (P ?x ?c)
Example: SAT – Fluents

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor \{N(x, c) \land \neg T(x)\}]\]

- \((T \ ?x)\) (not-T \ ?x)
- \((P \ ?x \ ?c)\)
- \((N \ ?x \ ?c)\)
Example: SAT – Fluents

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]\]

- (T ?x) (not-T ?x)
- (P ?x ?c)
- (N ?x ?c)
- (holds_and_6 ?x ?c)
Example: SAT – Fluents

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]\]

- (T ?x) (not-T ?x)
- (P ?x ?c)
- (N ?x ?c)
- (holds_and_6 ?x ?c)
- (holds_exists_8 ?c)
Example: SAT – Actions

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]\]

(:action set_true_11
 :parameters (?x0)
 :precondition (and (guess) (not_T ?x0))
 :effect (and (T ?x0) (not (not_T ?x0)))
)
Example: SAT – Actions

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]\]

(:action establish_and_6
  :parameters (?x ?c)
  :precondition (and (proof) (N ?x ?c) (not_T ?x))
  :effect (holds_and_6 ?x ?c)
)
Example: SAT – Actions

$$(\exists T)(\forall c)(\exists x)[(P(x,c) \land T(x)) \lor (N(x,c) \land \neg T(x)))]$$

(:action establish_exists_8
  :parameters (?x ?c)
  :precondition (and (proof) (holds_or_7 ?x ?c))
  :effect (holds_exists_8 ?x)
)
Example: SAT – Actions

\[(\exists T)(\forall c)(\exists x)[(P(x, c) \land T(x)) \lor (N(x, c) \land \neg T(x))]\]

(:action prove_forall_9_1
  :precondition (and (proof) (holds_exists_8 zero))
  :effect (holds_forall_9 zero)
)

(:action prove_forall_9_2
  :parameters (?y1 ?y2)
  :precondition (and (proof) (suc ?y1 ?y2)
                      (holds_forall_9 ?y1) (holds_exists_8 ?y2))
  :effect (holds_forall_9 ?y2)
)
Formal Properties

Let $\mathcal{G}(\text{dom}, \text{ins})$ be PDDL grounding function (generates STRIPS)

Define $f_{\sigma, \Phi} : \text{STRUC}[\sigma] \rightarrow \text{STRIPS}$ as

$$f_{\sigma, \Phi}(A) = \mathcal{G}(\mathcal{D}(\sigma, \Phi), \mathcal{I}(\sigma, \Phi, A))$$

where $\mathcal{D}(\sigma, \Phi)$ is the domain and $\mathcal{I}(\sigma, \Phi, A)$ is the instance.
Formal Properties

Let $\mathcal{G}(\text{dom}, \text{ins})$ be PDDL grounding function (generates STRIPS)

Define $f_{\sigma, \Phi} : \text{STRUC}[\sigma] \rightarrow \text{STRIPS}$ as

$$f_{\sigma, \Phi}(A) = \mathcal{G}(\mathcal{D}(\sigma, \Phi), \mathcal{I}(\sigma, \Phi, A))$$

**Thm:** $f_{\sigma, \Phi}$ is a polytime reduction from $\text{MOD}[\Phi]$ into a fragment of STRIPS that is decidable in NP
Let $\mathcal{G}(\text{dom}, \text{ins})$ be PDDL grounding function (generates STRIPS)

Define $f_{\sigma, \Phi} : \text{STRUC}[\sigma] \rightarrow \text{STRIPS}$ as

$$f_{\sigma, \Phi}(\mathcal{A}) = \mathcal{G}(\mathcal{D}(\sigma, \Phi), \mathcal{I}(\sigma, \Phi, \mathcal{A}))$$

\begin{align*}
\text{domain} & \quad \text{instance}
\end{align*}

**Thm:** $f_{\sigma, \Phi}$ is a polytime reduction from MOD[$\Phi$] into a fragment of STRIPS that is decidable in NP

**Thm:** if $f_{\sigma, \Phi}(\mathcal{A})$ has plan, it has one with parallel makespan at most $\text{MkSp}_\Phi(\mathcal{A}) = \mathcal{O}(\|\mathcal{A}\| \cdot \|\Phi\|)$ (i.e. linear in $\|\mathcal{A}\|$ for fixed $\Phi$)
Experiments
Performed in Xeon 1.86GHz CPUs with 2GB of RAM

Jussi Rintanen’s M planner (SAT-based planner)

Domains (all NP-Complete):
  - SAT
  - Clique
  - Directed Hamiltonian Paths
  - 3-Dimensional Matching
  - 3-Colorability
  - $k$-Colorability
  - Chromatic Number (beyond NP)

Instances:
  - SAT: from SATLIB w/ satisfiable and unsatisfiable instances
  - others: randomly generated w/ positive and negative instances
Summary of Results

- total of 1,920 problem instances
- total of 1,614 solved instances
  - 706 on positive side (input structure satisfies formula)
  - 908 on negative side (input structure doesn’t satisfy formula)
- M solved 84.06% of the benchmark
<table>
<thead>
<tr>
<th>Domains</th>
<th>$N^*/N$</th>
<th>#pos.</th>
<th>#neg.</th>
<th>avg. time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SAT</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uuf50</td>
<td>40/40</td>
<td>0</td>
<td>40</td>
<td>548.5</td>
</tr>
<tr>
<td>uuf75</td>
<td>1/40</td>
<td>0</td>
<td>1</td>
<td>1,746.4</td>
</tr>
<tr>
<td><strong>Clique</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25-3</td>
<td>40/40</td>
<td>30</td>
<td>10</td>
<td>111.9</td>
</tr>
<tr>
<td>25-4</td>
<td>40/40</td>
<td>18</td>
<td>22</td>
<td>231.0</td>
</tr>
<tr>
<td>25-5</td>
<td>39/40</td>
<td>10</td>
<td>29</td>
<td>387.5</td>
</tr>
<tr>
<td>25-6</td>
<td>36/40</td>
<td>8</td>
<td>28</td>
<td>394.1</td>
</tr>
<tr>
<td><strong>Hamiltonian Path</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>22/40</td>
<td>20</td>
<td>2</td>
<td>629.1</td>
</tr>
<tr>
<td><strong>3-dimensional Matching</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>13/40</td>
<td>13</td>
<td>0</td>
<td>1,191.0</td>
</tr>
<tr>
<td>25</td>
<td>0/40</td>
<td>0</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td><strong>3-colorability</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>40/40</td>
<td>1</td>
<td>39</td>
<td>196.7</td>
</tr>
</tbody>
</table>
### Chromatic Number

<table>
<thead>
<tr>
<th>instance</th>
<th>$\chi$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-0.75-1</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>101</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-0.75-2</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-0.85</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>1,265</td>
<td>4</td>
</tr>
<tr>
<td>15-0.25</td>
<td>2</td>
<td>27</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>15-0.60</td>
<td>5</td>
<td>27</td>
<td>29</td>
<td>54</td>
<td>118</td>
<td>72</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-0.70</td>
<td>6</td>
<td>28</td>
<td>28</td>
<td>33</td>
<td>47</td>
<td>329</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>20-0.10</td>
<td>3</td>
<td>214</td>
<td>350</td>
<td></td>
<td>705</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20-0.25</td>
<td>4</td>
<td>211</td>
<td>272</td>
<td>1,261</td>
<td>837</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Discussion

- tool produces PDDLs from declarative descriptions
- can be thought as automatic generation of reductions
- different translations available, only one implemented
- different applications for the tool
- not every NP problem has a nice formula!

Future:

- improve tool by incorporating types, other translations, . . .
- aim at other complexity classes (fragments of logic)
Thanks!