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

Descriptive Complexity Theory

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$\exists$)

Main results of DCT:

- P equals SO-Horn
- NP equals SO$\exists$ and coNP equals SO$\forall$
- 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 $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause 0}} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause 1}} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause 2}}$

Example: SAT

$$\text{CNF } \varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause } 0} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause } 1} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause } 2}$$

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 $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause } 0} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause } 1} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause } 2}$

E.g., φ 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

CNF $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause 0}} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause 1}} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause 2}}$

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 $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause 0}} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause 1}} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause 2}}$

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

$$(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \vee (N(x,c) \wedge \neg T(x))]$$

iff T encodes a model of φ

Example: SAT

CNF $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause 0}} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause 1}} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause 2}}$

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

$$(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \vee (N(x,c) \wedge \neg T(x))]$$

iff T encodes a model of φ

Hence, φ is SATISFIABLE iff $\mathcal{A}$ is model of

$$\Phi = \underbrace{(\exists T)}_{\text{s.o.}\exists}(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \vee (N(x,c) \wedge \neg T(x))]$$

Example: SAT

CNF $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause 0}} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause 1}} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause 2}}$

Indeed, SAT $=$ MOD$[\Phi]$

Example: SAT

CNF $\varphi = \underbrace{(x_0 \vee \neg x_1 \vee x_2)}_{\text{clause 0}} \wedge \underbrace{(\neg x_0 \vee \neg x_2)}_{\text{clause 1}} \wedge \underbrace{(\neg x_0 \vee x_1)}_{\text{clause 2}}$

Indeed, SAT $=$ MOD$[\Phi]$

Meaning:

- for every satisfiable formula φ, its encoding $\mathcal{A}_\varphi \in$ MOD$[\Phi]$
- for every $\mathcal{A} \in$ MOD$[\Phi]$, $\mathcal{A}$ encodes a satisfiable formula $\varphi_\mathcal{A}$

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

$$
\begin{aligned}
(\exists R^1, G^1, B^1)(\forall x, y)[\\
&(R(x) \vee G(x) \vee B(x)) \wedge \\
&R(x) \rightarrow \neg(G(x) \vee B(x)) \wedge \\
&G(x) \rightarrow \neg(R(x) \vee B(x)) \wedge \\
&B(x) \rightarrow \neg(R(x) \vee G(x)) \wedge \\
&E(x, y) \rightarrow \neg[(R(x) \wedge R(y)) \vee (G(x) \wedge G(y)) \vee (B(x) \wedge B(y))]\,]
\end{aligned}
$$

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 $F :: [0...n] \longrightarrow |\mathcal{A}|$

$$(\exists F \in \mathsf{Inj})(\forall x)[x < \max \rightarrow (\exists x' y z)(E(y, z) \wedge F(x, y) \wedge \mathsf{SUC}(x, x') \wedge F(x', z))]$$

The Tool

$$\xrightarrow[\sigma,\Phi,\mathcal{A}]{\text{DCT}} \boxed{\text{tool}} \xrightarrow[\text{dom \& ins}]{\text{PDDL}}$$

Input:

- signature σ that contains relational symbols
- SO$\exists$ formula Φ that encodes NP problem
- finite structure $\mathcal{A}$ that encodes instance

Output:

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

Guarantees:

- runs in polytime for fixed Φ
- output is no harder that input (complexity wise)

Related Work

- `DATALOG`-like specification of NP problems into SAT (Cadoli & Schaerf, 2005)
  - we are targetting 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:

$$\mathfrak{D} : \text{Signatures} \times \text{SO}\exists \longrightarrow \text{PDDL Domains}$$

$$\mathfrak{I} : \text{Signatures} \times \text{SO}\exists \times \text{STRUC} \longrightarrow \text{PDDL Instances}$$

Different Translations

Translations that aim different planners:

- for sequential planners
- for parallel planners
- for optimal sequential planners

Translation Used in Experiments

Domain $\mathfrak{D}(\sigma, \Phi)$:

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

Translation Used in Experiments

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

- 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

$$\underbrace{(\exists T)}(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \;\vee\; (N(x,c) \wedge \neg T(x))]$$

- `(T ?x) (not-T ?x)`

Example: SAT – Fluents

$$(\exists T)(\forall c)(\exists x)[(\underbrace{P(x,c)} \wedge T(x)) \;\vee\; (N(x,c) \wedge \neg T(x))]$$

- `(T ?x) (not-T ?x)`
- `(P ?x ?c)`

Example: SAT – Fluents

$$(\exists T)(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \ \vee \ (\underbrace{N(x,c)} \wedge \neg T(x))]$$

- `(T ?x) (not-T ?x)`
- `(P ?x ?c)`
- `(N ?x ?c)`

Example: SAT – Fluents

$$(\exists T)(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \ \vee \ \underbrace{(N(x,c) \wedge \neg T(x))}]$$

- `(T ?x) (not-T ?x)`
- `(P ?x ?c)`
- `(N ?x ?c)`
- `(holds_and_6 ?x ?c)`

Example: SAT – Fluents

$$(\exists T)(\forall c)\underbrace{(\exists x)[(P(x,c) \wedge T(x)) \;\vee\; (N(x,c) \wedge \neg T(x))]}$$

- `(T ?x) (not-T ?x)`
- `(P ?x ?c)`
- `(N ?x ?c)`
- `(holds_and_6 ?x ?c)`
- `(holds_exists_8 ?c)`

Example: SAT – Actions

$$\underbrace{(\exists T)}(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \ \vee \ (N(x,c) \wedge \neg T(x))]$$

```
(:action set_true_11
  :parameters    (?x0)
  :precondition  (and (guess) (not_T ?x0))
  :effect        (and (T ?x0) (not (not_T ?x0)))
)
```

Example: SAT – Actions

$$(\exists T)(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \;\vee\; \underbrace{(N(x,c) \wedge \neg T(x))}]$$

```
(: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

$$(\exists T)(\forall c)\,\underbrace{(\exists x)[(P(x,c) \wedge T(x)) \;\vee\; (N(x,c) \wedge \neg T(x))]}$$

```
(:action establish_exists_8
  :parameters    (?x ?c)
  :precondition  (and (proof) (holds_or_7 ?x ?c))
  :effect        (holds_exists_8 ?x)
)
```

Example: SAT – Actions

$$(\exists T)\,\underbrace{(\forall c)(\exists x)[(P(x,c) \wedge T(x)) \ \vee \ (N(x,c) \wedge \neg T(x))]}$$

```
(: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 $\mathfrak{G}(\mathrm{dom}, \mathrm{ins})$ be PDDL grounding function (generates STRIPS)

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

$$f_{\sigma,\Phi}(\mathcal{A}) = \mathfrak{G}(\underbrace{\mathfrak{D}(\sigma, \Phi)}_{\text{domain}}, \underbrace{\mathfrak{I}(\sigma, \Phi, \mathcal{A})}_{\text{instance}})$$

Formal Properties

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

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

$$f_{\sigma,\Phi}(\mathcal{A}) = \mathfrak{G}(\,\underbrace{\mathfrak{D}(\sigma,\Phi)}_{\text{domain}},\ \underbrace{\mathfrak{I}(\sigma,\Phi,\mathcal{A})}_{\text{instance}}\,)$$

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

Formal Properties

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

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

$$f_{\sigma,\Phi}(\mathcal{A}) = \mathfrak{G}(\underbrace{\mathfrak{D}(\sigma, \Phi)}_{\text{domain}}, \underbrace{\mathfrak{I}(\sigma, \Phi, \mathcal{A})}_{\text{instance}})$$

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 Φ)

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

Domains

	N^*/N	#pos.	#neg.	avg. time
SAT				
uuf50	40/40	0	40	548.5
uuf75	1/40	0	1	1,746.4
Clique				
25-3	40/40	30	10	111.9
25-4	40/40	18	22	231.0
25-5	39/40	10	29	387.5
25-6	36/40	8	28	394.1
Hamiltonian Path				
30	22/40	20	2	629.1
3-dimensional Matching				
20	13/40	13	0	1,191.0
25	0/40	0	0	—
3-colorability				
50	40/40	1	39	196.7

Chromatic Number

		k-colorability						
instance	χ	1	2	3	4	5	6	7
10-0.75-1	5	2	2	6	101	**3**		
10-0.75-2	5	1	2	2	6	**4**		
10-0.85	7	2	2	3	6	4	1,265	**4**
15-0.25	2	27	**62**					
15-0.60	5	27	29	54	118	**72**		
15-0.70	6	28	28	33	47	329	**67**	
20-0.10	3	214	350	**705**				
20-0.25	4	211	272	1,261	**837**			

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!