A Comparison of Invariant Synthesis Methods

Severin Wyss

University of Basel

November 5, 2020

Introduction

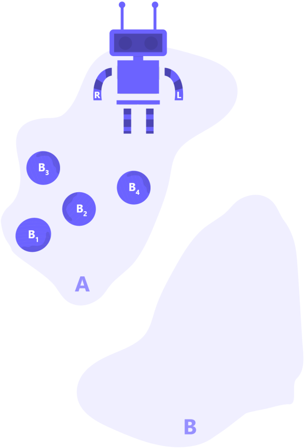
Motivation

Motivation

- problems generally modeled in PDDL
- Fast Downward (fd) uses finite domain representation (FDR)
- PDDL to FDR requires Mutex groups
- fd uses Monotonicity Invariants for Mutex groups[Helmert(2009)]
- other methods exist
- compare Rintanen's algorithm (G-IRIS) to fd's algorithm (S-MIS)

Running Example: Gripper

- a robot with 2 arms (grippers): R, L
- 4 balls: b_1, b_2, b_3, b_4
- 2 rooms: A, B
- balls in rooms: $\text{in}(b_1, A)$
- or carried: $\text{carry}(b_1, L)$
- grippers can be empty: $\text{free}(L)$
- robot in rooms: $\text{at-robby}(A)$
- initial state: balls and robot in A
- goal: all balls in B



Running Example: Gripper

$$
\begin{aligned}
move(A, B) = \quad & \langle at\text{-}robby(A), \\
& \{\neg at\text{-}robby(A), at\text{-}robby(B)\}\rangle \\
\ldots \\
pick\text{-}up(A, b_1, L) = & \langle at\text{-}robby(A) \land at(b_1, A) \land free(L), \\
& \{\neg at(b_1, A), \neg free(L), carry(b_1, L)\}\rangle \\
\ldots \\
drop(B, b_1, L) = \quad & \langle at\text{-}robby(B) \land carry(b_1, L), \\
& \{\neg carry(b_1, L), at(b_1, B), free(L)\}\rangle \\
\ldots
\end{aligned}
$$

Background

Background

Invariants

- invariants are formulas
- that hold in every reachable state

i.e. at-robby(A) $\vee$ at-robby(B)

Background

Mutex

- a mutex is an invariant with only contains negative literals.
- for us explicitly: mutex have length 2.

i.e. $\neg \text{in}(b_1, \text{A}) \vee \neg \text{in}(b_1, \text{B})$

Background

Mutex Group

- mutex group is a set of atoms
- exist mutex to each pair
- only ever one atom of a mutex group can be true at once.

i.e. $\{\text{in}(b_1, \text{A}), \text{in}(b_1, \text{B}), \text{carry}(b_1, \text{L}), \text{carry}(b_1, \text{R})\}$

Background

Mutex Groups to FDR

- mutex group $\rightarrow$ finite domain representation variable
- FDR-variable has domain = mutex group
- adapt rest of planning task

i.e. the mutex group $\{\text{in}(b_1, \text{A}), \text{in}(b_1, \text{B}), \text{carry}(b_1, \text{L}), \text{carry}(b_1, \text{R})\}$
leads to v with $d(v) = \{\text{in}(b_1, \text{A}), \text{in}(b_1, \text{B}), \text{carry}(b_1, \text{L}), \text{carry}(b_1, \text{R})\}$

G-IRIS

G-IRIS

Name

G-IRIS : ground - iterative reachability invariant synthesis

Reachability Invariant Candidates (RIC)

a disjunction of literals

$$\gamma_1 = (l_1 \vee l_2)$$
$$\gamma_2 = l_1$$

G-IRIS

Rintanens Algorithm (G-IRIS)

Algorithm 1: Concept of G-IRIS

```
function G-IRIS(...):
    initialize(Γ)
    while Γ ≠ Γ' do
        Γ' := Γ
        foreach a ∈ A and γ ∈ Γ s.t. reachable(¬γ, Γ') do
            Γ := Γ\{γ}
            weaken(γ)
    return Γ
```

G-IRIS

G-IRIS: Input

- G: a finite set of ground atoms
- s_0: the initial state
- A: a finite set of grounded actions
- $n \in \mathbb{N}$

G-IRIS

G-IRIS: initialize(Γ)

$$\Gamma := \{g \in G \mid s_0 \models g\} \cup \{\neg g \mid g \in G; s_0 \not\models g\}$$

- start with initial state
- could be invariants
- also explicit negative literals
- negation cannot be invariant → all valid candidates of size 1

G-IRIS

G-IRIS: Regression of Negation of RIC

$$rg_a(\neg\gamma)$$

If an action a falsifies the candidate γ, then γ cannot be a invariant if a is applicable a in reachable state.
regression in STRIPS: $rg_a(s) = \chi \wedge \psi$

- χ is precondition of a
- ψ is γ minus the effect of a

i.e. $\gamma = \neg \textit{at-robby}(B)$
$\neg\gamma = \textit{at-robby}(B)$
$a = \textit{move}(A, B) = \langle \textit{at-robby}(A), \{\neg \textit{at-robby}(A), \textit{at-robby}(B)\}\rangle$
$rg_a(s) = \textit{at-robby}(A)$

G-IRIS

G-IRIS: reachable($\neg\gamma, \Gamma'$)

$$\Gamma' \cup \{rg_a(\neg\gamma)\} \in \text{SAT}$$

we test reachability with the current set of candidates.
In first iteration that is the initial state.
Afterwards, candidates not disproved by the previous set.

G-IRIS

G-IRIS: weaken(γ)

IF $|\texttt{lits}\ (\gamma)| < n$ then

$$\Gamma := \Gamma \cup \{\gamma \vee g \mid g \in G\} \cup \{\gamma \vee \neg g \mid g \in G\}$$

lits(c) returns the number of atoms in the formula.
Candidate size limited ($n = 2$ in our case) due to runtime (more on that later).
Weaken by creating disjunctions with all facts and their negations.

G-IRIS

Rintanens Algorithm (G-IRIS)

Algorithm 2: G-IRIS

Input: G, s_0, A and n.

Output: Γ (set of RIC proven invariant)

function `invariants`(G, s_0, A, n):

$\Gamma := \{g \in G \mid s_0 \models g\} \cup \{\neg g \mid g \in G; s_0 \not\models g\}$

while $\Gamma \neq \Gamma'$ **do**

$\Gamma' := \Gamma$

foreach $a \in A$ *and* $\gamma \in \Gamma$ *s.t.* $\Gamma' \cup \{rg_a(\neg\gamma)\} \in SAT$ **do**

$\Gamma := \Gamma \setminus \{\gamma\}$

if $|\texttt{lits}\ (\gamma)| < n$ **then**

$\Gamma := \Gamma \cup \{\gamma \vee g \mid g \in G\} \cup \{\gamma \vee \neg g \mid g \in G\}$

return Γ

G-IRIS

G-IRIS Example

$\Gamma_0 = \{\mathrm{at}(b_1, A), ..., \mathrm{free}(L), ..., \neg\mathrm{carry}(b_1, L), ..., \neg\mathrm{at}(b_1, B), ...\}$
$\Gamma_1 = \{..., \neg\mathrm{carry}(b_1, L) \vee \neg\mathrm{carry}(b_2, R), ..., \neg\mathrm{at}(b_1, B), ...\}$
$\Gamma_2 = \{..., \neg\mathrm{at}(b_1, B), ...\}$
$\Gamma_3 = \{..., \neg\mathrm{at}(b_1, A) \vee \neg\mathrm{at}(b_1, B), ...\}$

G-IRIS

G-IRIS

Important: invariants only proven when fix point reached

G-IRIS

G-IRIS: Output

$\neg$at(b_1, A) $\vee$ $\neg$carry(b_1, R),
$\neg$carry(b_1, R) $\vee$ $\neg$carry(b_4, R),
...
$\neg$at-robby(A) $\vee$ $\neg$at-robby(B),
...
$\neg$carry(b_2, *left*) $\vee$ $\neg$free(*left*),
at-robby(A) $\vee$ at-robby(B),
...
$\neg$free(R) $\vee$ free(R),
...

G-IRIS

G-IRIS: Building Mutex Group

1. Select form $\neg a \vee \neg b$
2. Build graph
   1. Nodes = Atoms
   2. Edges (n_0, n_1) and (n_1, n_0) for $\neg n_0 \vee \neg n_1$
3. greedy clique algorithm i.e. by Rintanen[Rintanen(2006)]
4. mutex groups = cliques

Implementation G-IRIS: SAT by not unsat

algo works for testing not unsat.
we implemented only for size 2 and could therefore use 2-sat

Implementation

Optimization G-IRIS: Trivial Candidates

correct but uninformative invariants $a \vee \neg a$ (tautology)
the implementation handles them in a separate list to reduce the inner loop

Implementation

Optimization G-IRIS: Weaken

the resulting weakened candidates are immediately tested against the same operator
test: did weakening solved reason for rejection
candidates are only tested against operators that affect candidate

Evaluation

Evaluation of Translation

Problem Domains

Two tracks of STRIPS problems from IPCs

- optimal track (opt), 1133 problems in 35 domains
- satisficing track (sat), 1119 problems in 34 domains

Evaluation of Translation

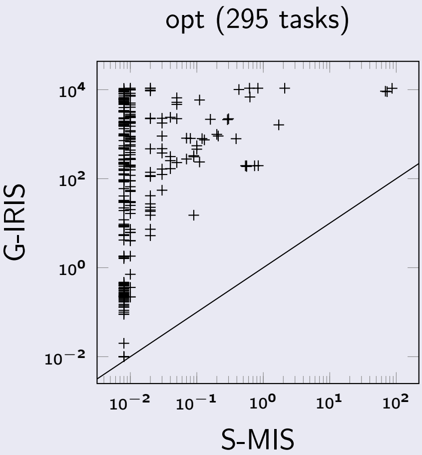
Number of Problems Translated

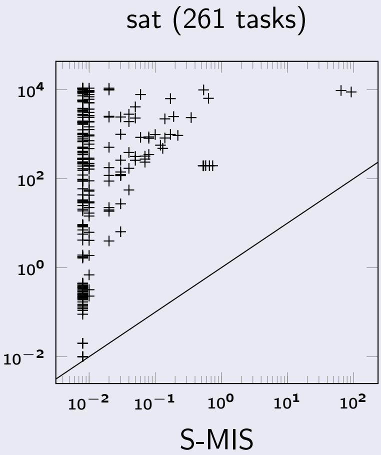
Table: translated by G-IRIS and S-MIS (fd)

		opt		sat	
algorithm	limit	done	out of time	done	out of time
S-MIS	5m	1133	0	1119	0
G-IRIS	5m	151	982	140	979
G-IRIS	3h	295	838	261	858

Evaluation of Translation

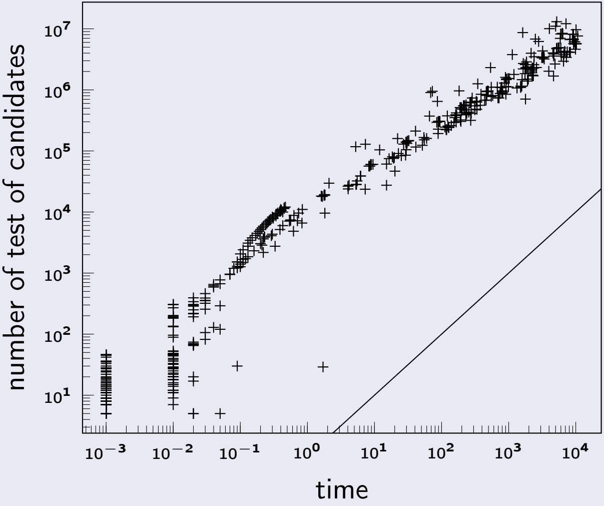
Runtime G-IRIS vs. S-MIS





Evaluation of Translation

Explaining Runtime of G-IRIS



Evaluation of Search

Search using A^*

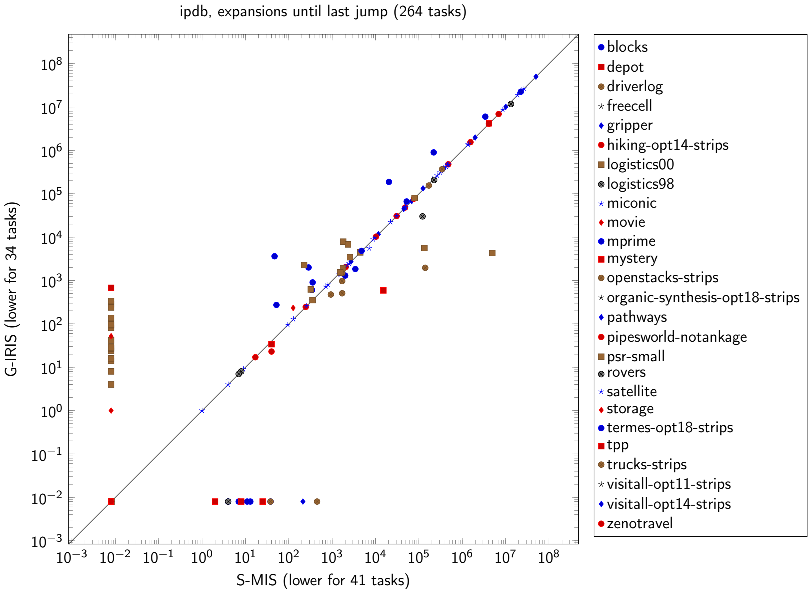
Table: solved in opt

solved by	blind	hmax	ipdb	lmcut	m&s
only S-MIS	0	2	23	25	4
both	267	269	264	259	285
only G-IRIS	0	0	0	0	0

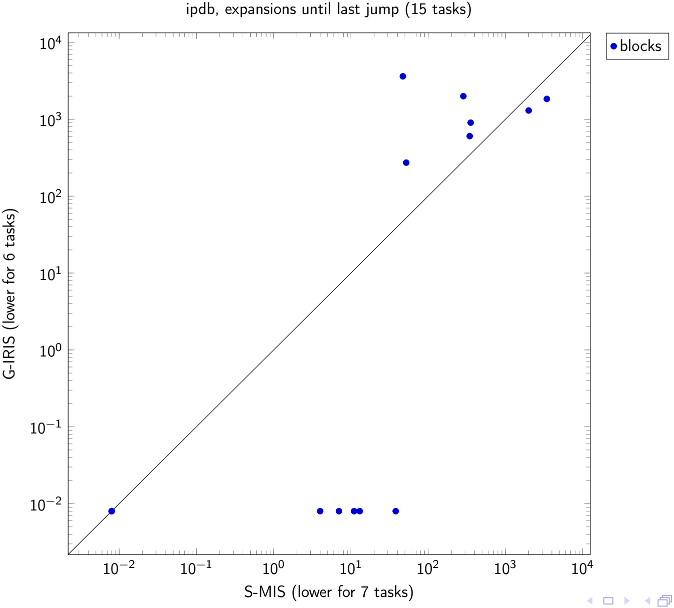
Table: solved in sat

solved by	blind	hmax	ipdb	lmcut	m&s
only S-MIS	0	0	16	19	1
both	237	237	235	234	253
only G-IRIS	0	0	0	0	0

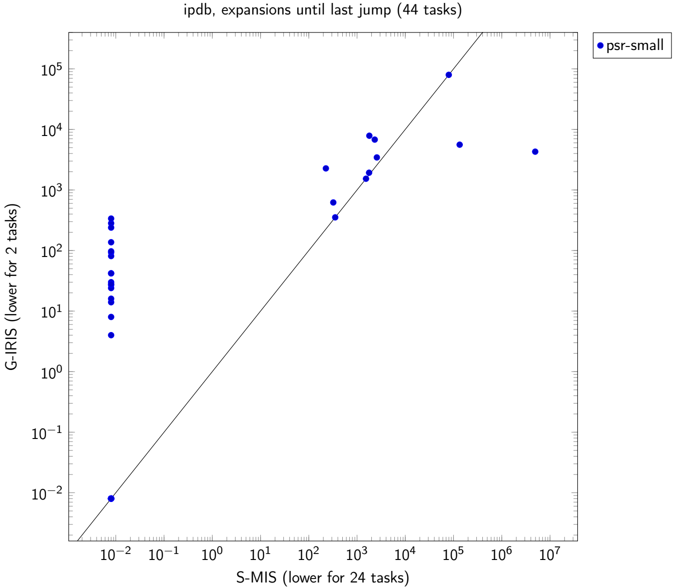
Evaluation of Search



Evaluation of Search



Evaluation of Search



Evaluation of Search

Search: Further Characteristics

Compared translation with regard to:

- number of binary variables
- number of variables
- number of facts
- number of actions

No conclusive results.

Evaluation

Summary

- G-IRIS can only translate about 13% problems tested within 5 minutes
- G-IRIS more than 100 times slower than S-MIS
- Translation by G-IRIS could be solved fewer times
- Translated and solved by both: no consistent benefit to either

S-IRIS

S-IRIS

S-IRIS: Observation

Same invariants for

- b_1
- b_2
- b_3
- b_4

How many are necessary?

$\neg\text{in}(b_1, \text{A}) \vee \neg\text{in}(b_2, \text{A})$

for $n = 2$ at least 2

S-IRIS

S-IRIS: Limited Grounding

minimal necessary objects of types from [Rintanen(2017)]

- candidates (*n* and predicates)
- actions

S-IRIS

S-IRIS: Schematic Version of G-IRIS

1. limited grounding: planning task with only minimal objects
2. solve with G-IRIS
3. infer invariants for other objects

S-IRIS

S-IRIS: Translation Evaluation

Table: translated by S-IRIS

		depot		logistics98	
algorithm	limit	done	out of time	done	out of time
G-IRIS	3h	2	20	2	33
S-IRIS	3h	22	0	22	13

Summary

Summary

Summary

- G-IRIS: fix point iteration, regression of negation
- G-IRIS slower than S-MIS
- fewer translation by G-IRIS where solved
- S-IRIS uses less objects, therefore faster
- IRIS can not stop early
- translations by IRIS do not deliver consistent benefit.

References I

Malte Helmert.
Concise finite–domain representations for PDDL planning tasks.
Artificial Intelligence, 173(5–6):503–535, 2009.

Jussi Rintanen.
Compact representation of sets of binary constraints.
In *ECAI*, volume 141, pages 143–147, 2006.

Jussi Rintanen.
Schematic invariants by reduction to ground invariants.
In *Thirty–First AAAI Conference on Artificial Intelligence*, pages 3644–3650, 2017.

The End

Thank you for your attention.

Questions

Please fire away with your inputs and questions.

Questions

Clique Computation

- influence of max cliques on search?
- smarter greedy algorithm?
- maybe adapt fast downwards mutex group selection?

Questions

$n > 2$

- increase runtime
- can find more invariants ($a \vee b \vee c$)
- not helpful for mutex groups

Questions

PDDL without Types

- not all problems use pddl types
- for exmaple gripper
- however: unary static predicates imply types
- implemented translation preceeding S-IRIS

Questions

RIC

ϕ is a disjunction of literals and ψ is a (possibly empty) conjunction of inequalities $x \neq x'$ where x and x' are objects or variables

$$\gamma_0 = \psi_0 \rightarrow (l_1 \vee l_2)$$
$$\gamma_1 = \psi_1 \rightarrow l_1$$

S-IRIS with Schematic Candidates

limited grounding possible at several points (outer loop, on demand)
weakening more complex:

- add literal
- partially ground
- change inequalities

candidates imply set of ground candidates

A Comparison of Invariant Synthesis Methods

Severin Wyss

University of Basel

November 5, 2020