Universal Planning in Non-Deterministic Domains

Rune M. Jensen

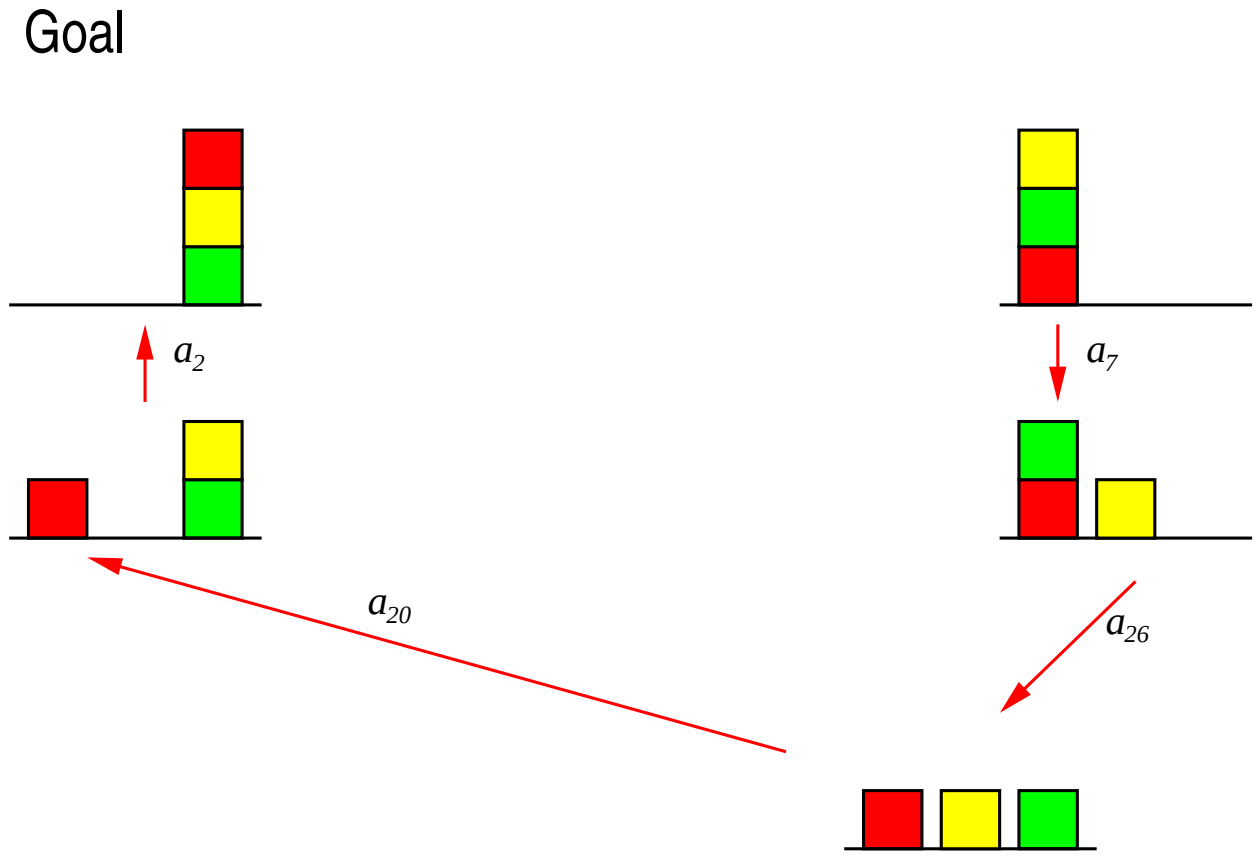
www.cs.cmu.edu/∼runej

Computer Science Department

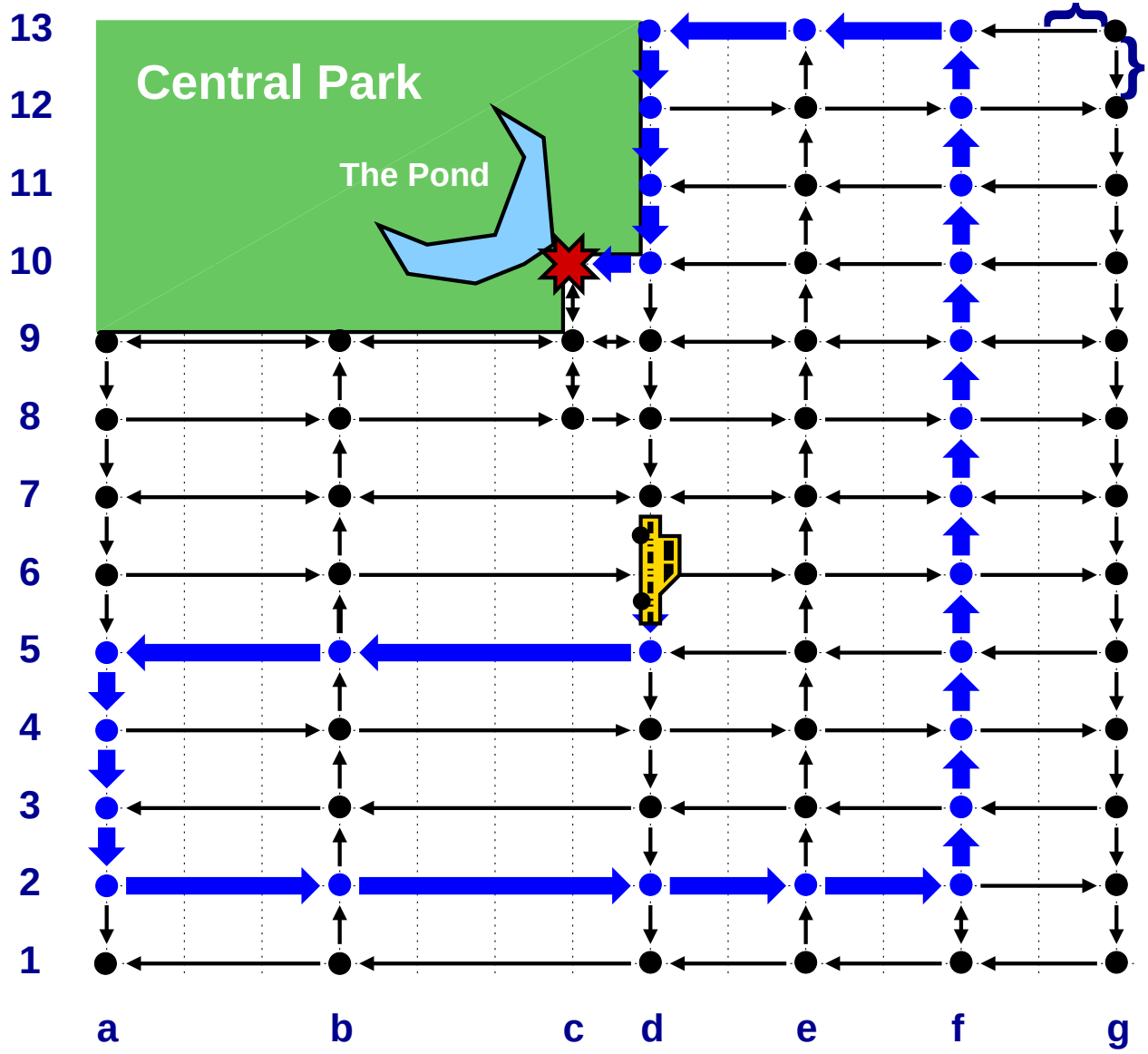
Carnegie Mellon University

USA

Planning in Deterministic Domains

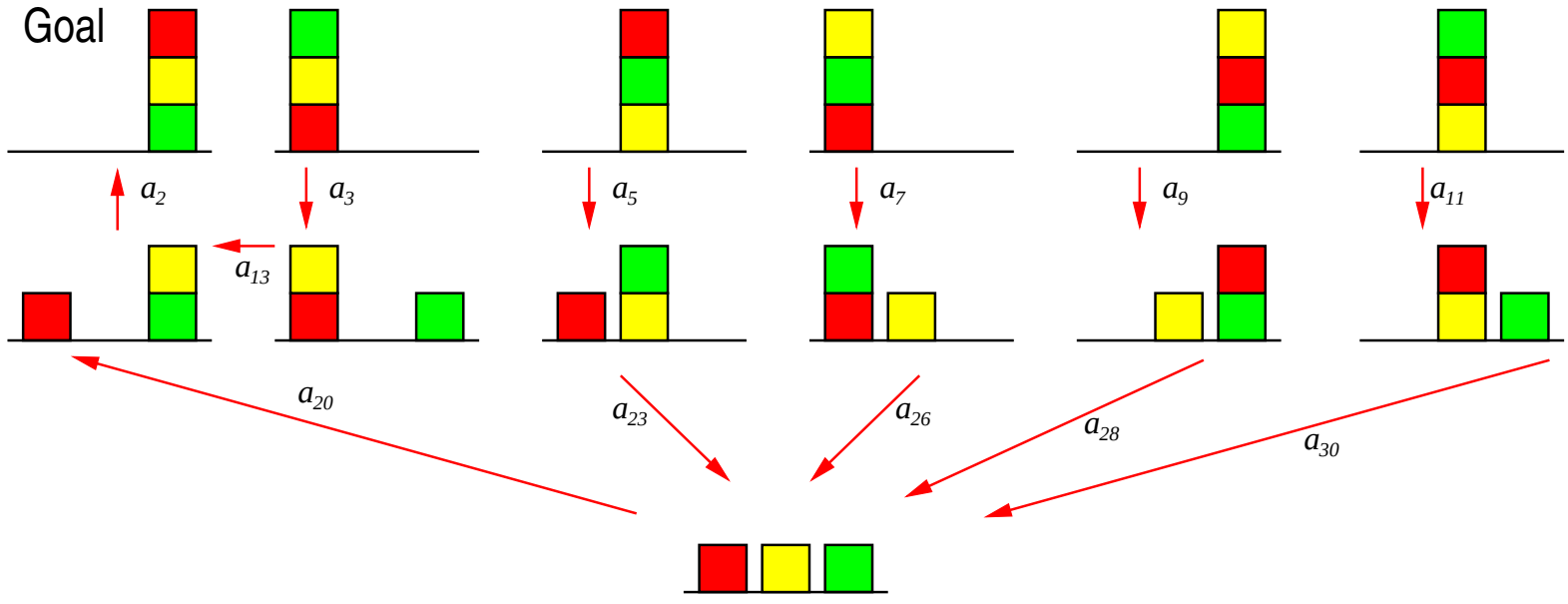


Planning in Deterministic Domains



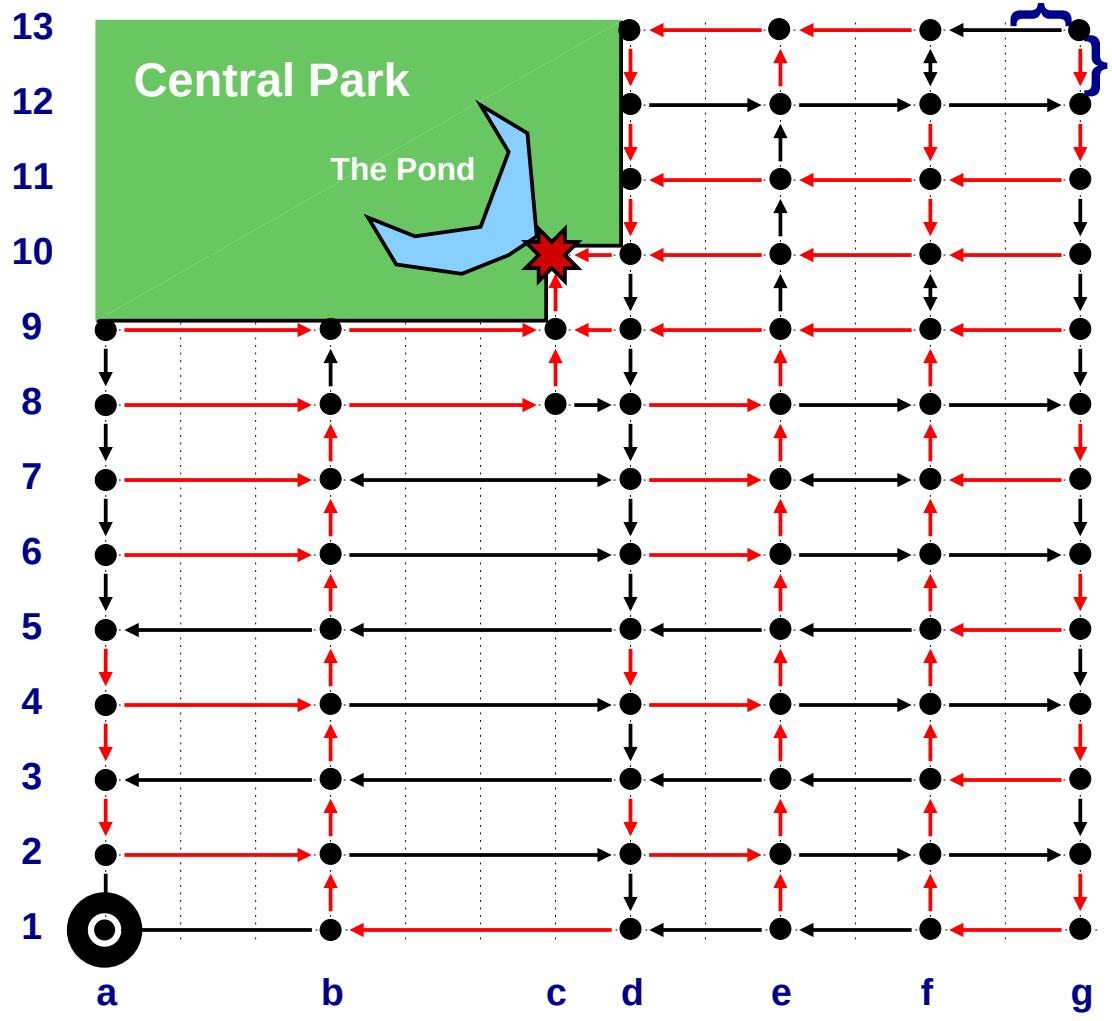
Universal Planning

In Deterministic Domains



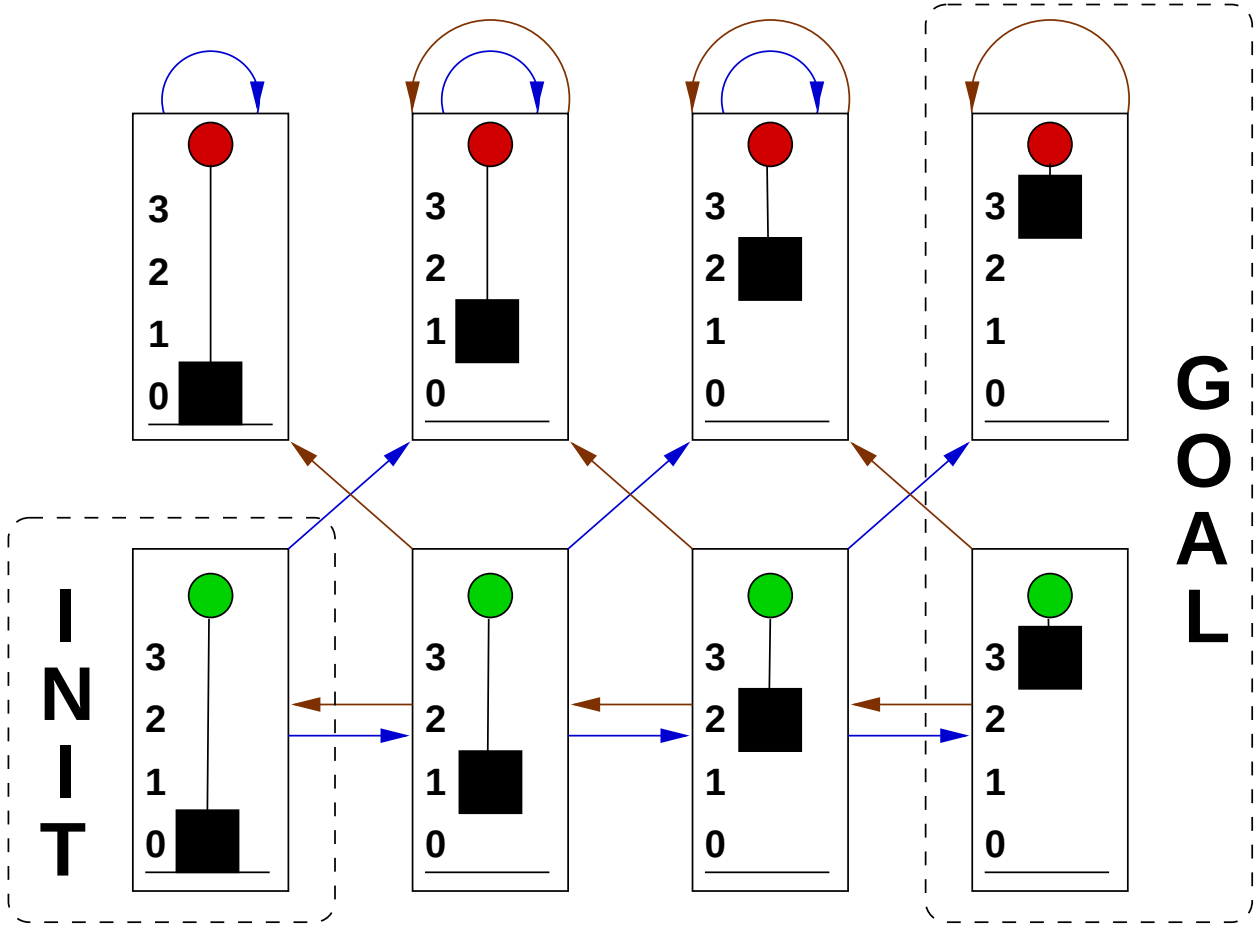
Universal Planning

In Deterministic Domains



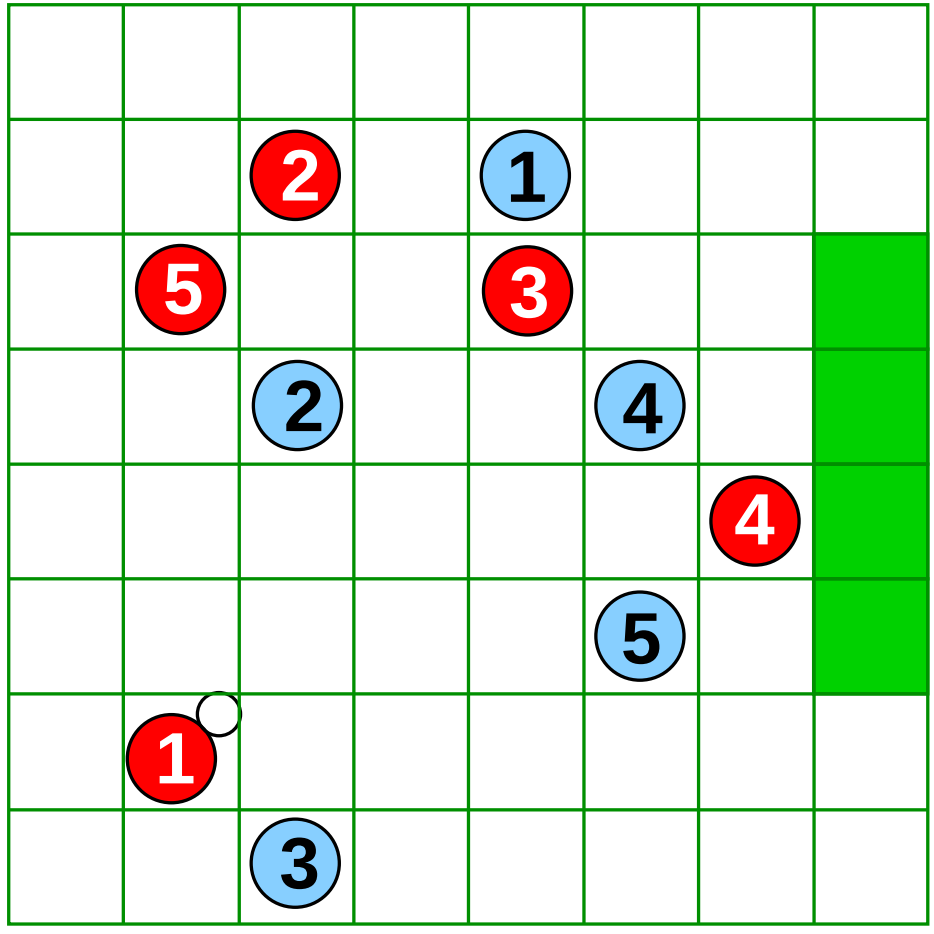
Universal Planning

In Non-Deterministic Domains



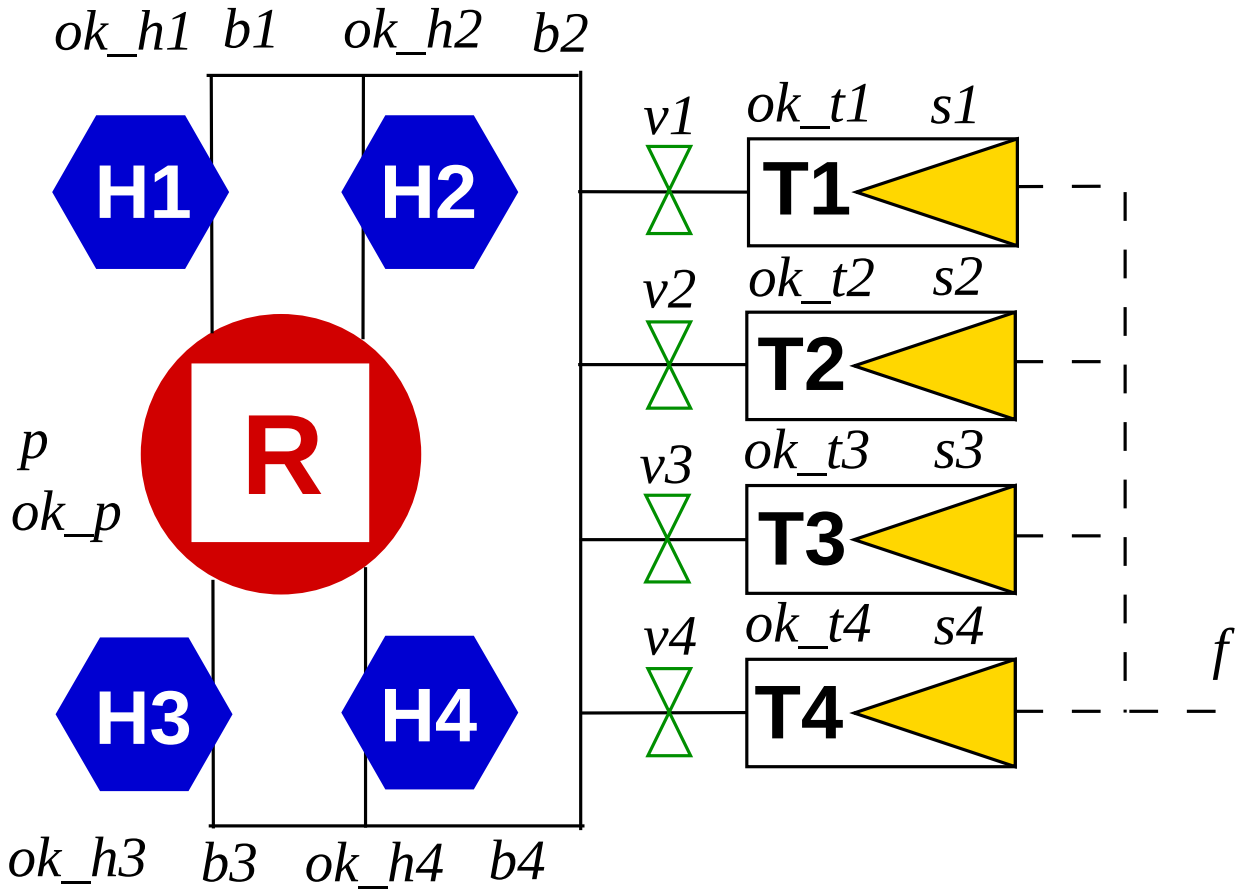
Universal Planning

In Non-Deterministic Domains



Universal Planning

In Non-Deterministic Domains



What is Universal

about a Universal Plan ?

- It covers all states in the domain such that uncontrollable effects from the environment can be handled (Schoppers 87)

What is Universal

about a Universal Plan ?

- It covers all states in the domain such that uncontrollable effects from the environment can be handled (Schoppers 87)
- It covers all states in any possible execution trace from a set of initial states (Cimatti et al. 98)

What is Universal

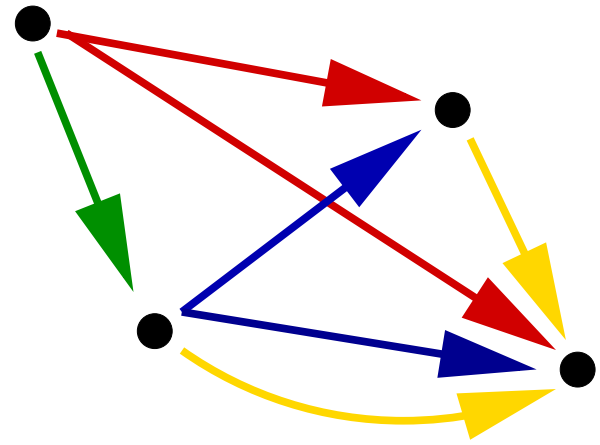
about a Universal Plan ?

- It covers all states in the domain such that uncontrollable effects from the environment can be handled (Schoppers 87)
- It **ideally** covers all states in any possible execution trace from a set of initial states (Cimatti et al. 98)

Definition of a Universal planning problem

We need:

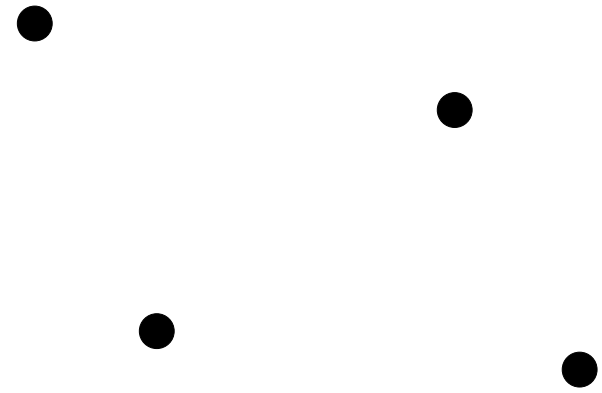
1. A non-deterministic transition system



Definition of a Universal planning problem

We need:

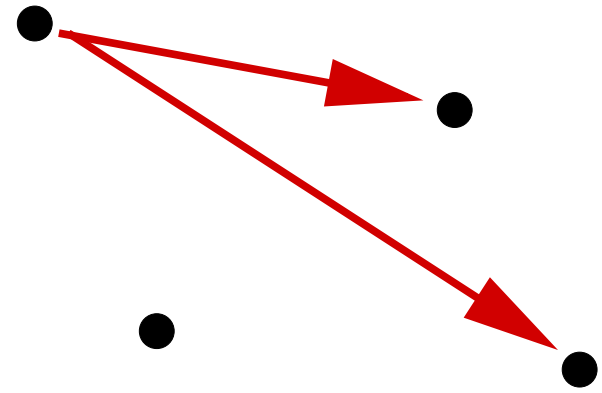
1. A non-deterministic transition system



Definition of a Universal planning problem

We need:

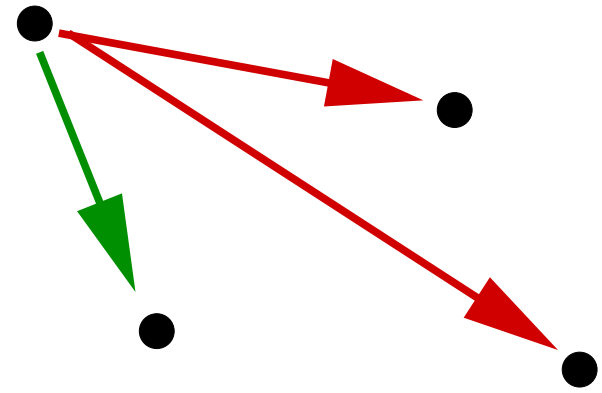
1. A non-deterministic transition system



Definition of a Universal planning problem

We need:

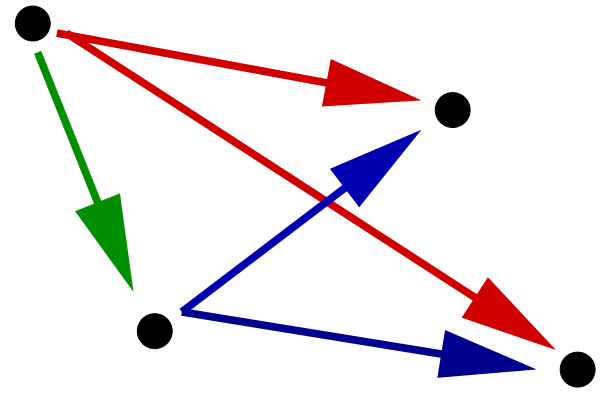
1. A non-deterministic transition system



Definition of a Universal planning problem

We need:

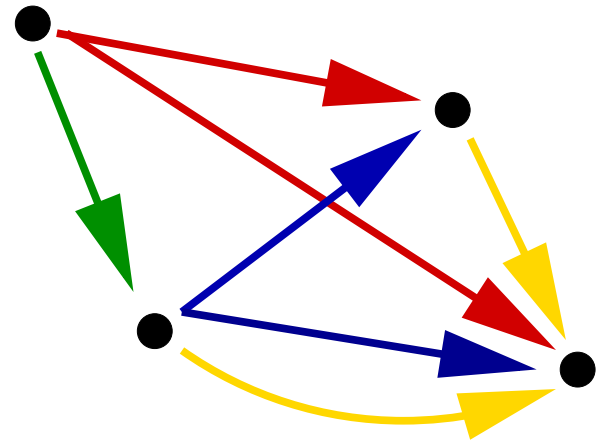
1. A non-deterministic transition system



Definition of a Universal planning problem

We need:

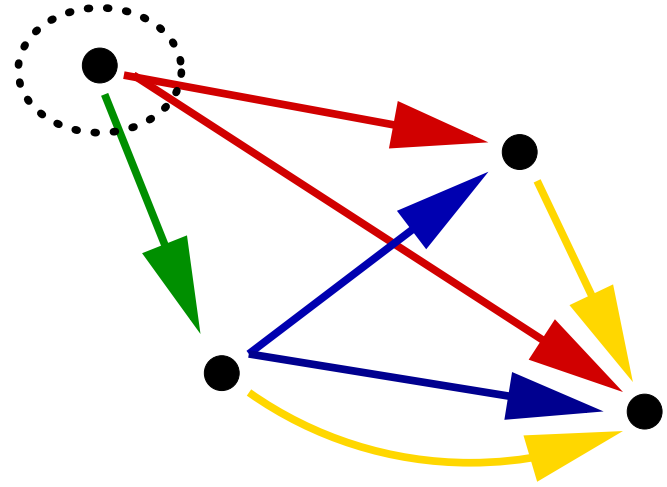
1. A non-deterministic transition system



Definition of a Universal planning problem

We need:

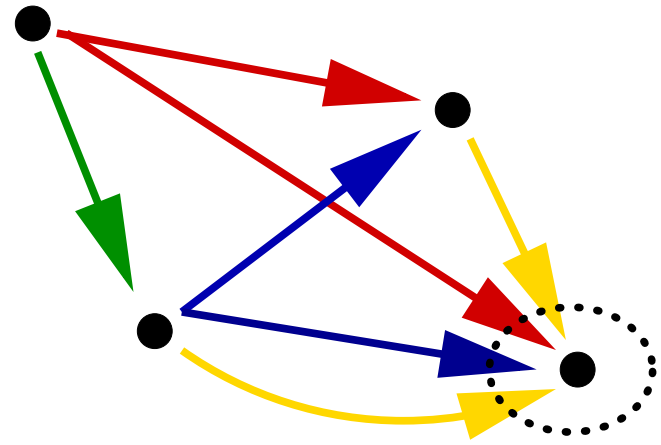
1. A non-deterministic transition system
2. A set of initial states



Definition of a Universal planning problem

We need:

1. A non-deterministic transition system
2. A set of initial states
3. A set of goal states

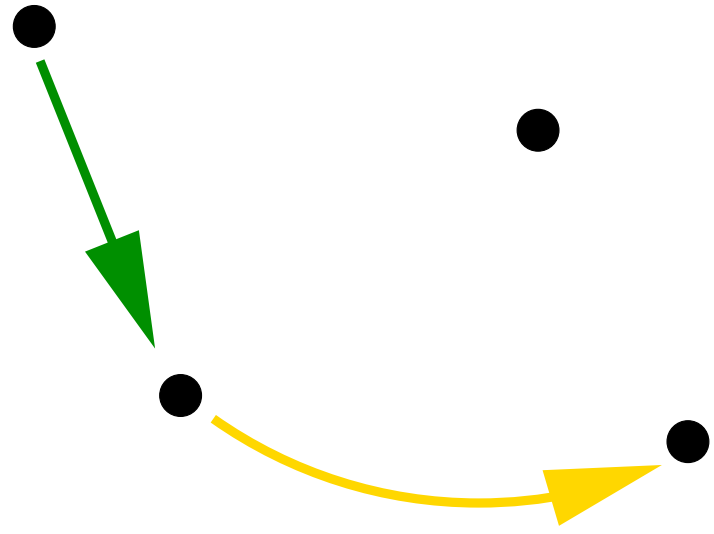


Definition of a Universal Plan

A universal plan is **state-action table**

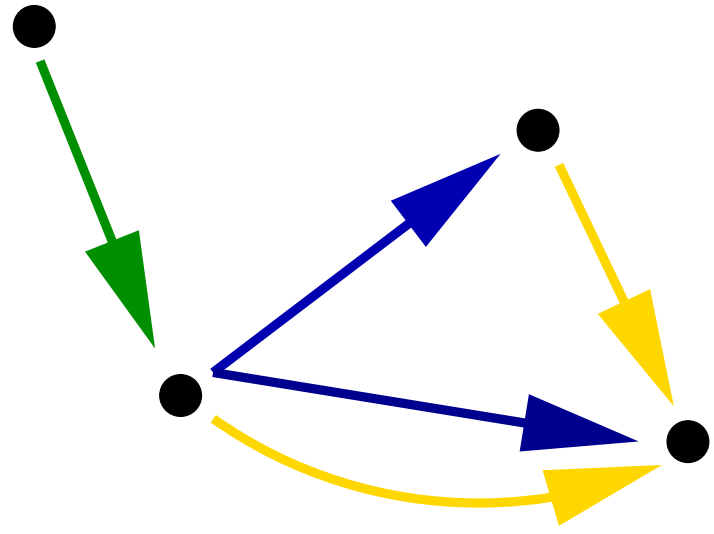
Definition of a Universal Plan

A universal plan is **state-action table**



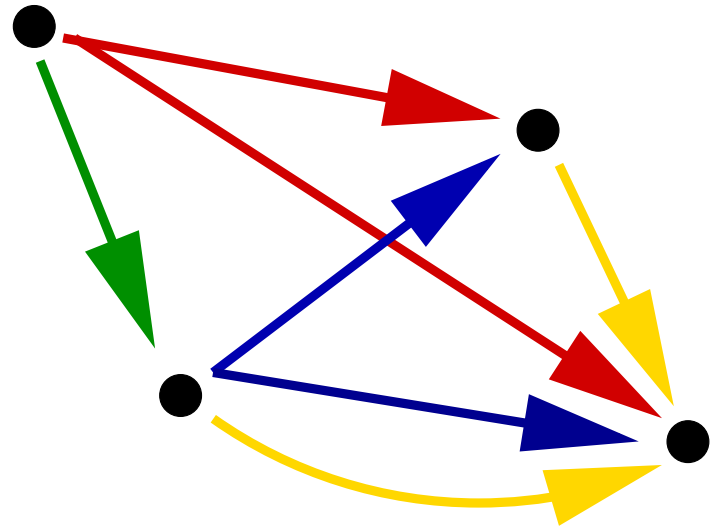
Definition of a Universal Plan

A universal plan is **state-action table**



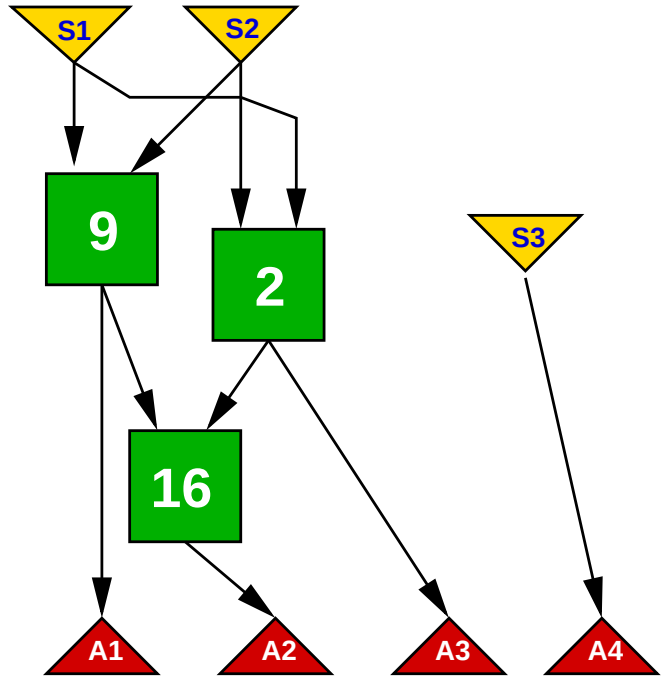
Definition of a Universal Plan

A universal plan is **state-action table**



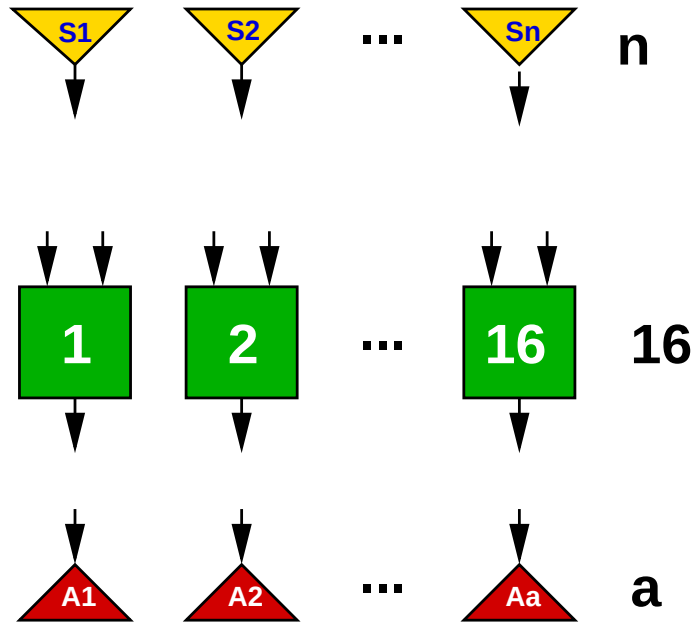
Ginsberg Critique (Ginsberg 89)

Universal plans are impossible to represent efficiently !



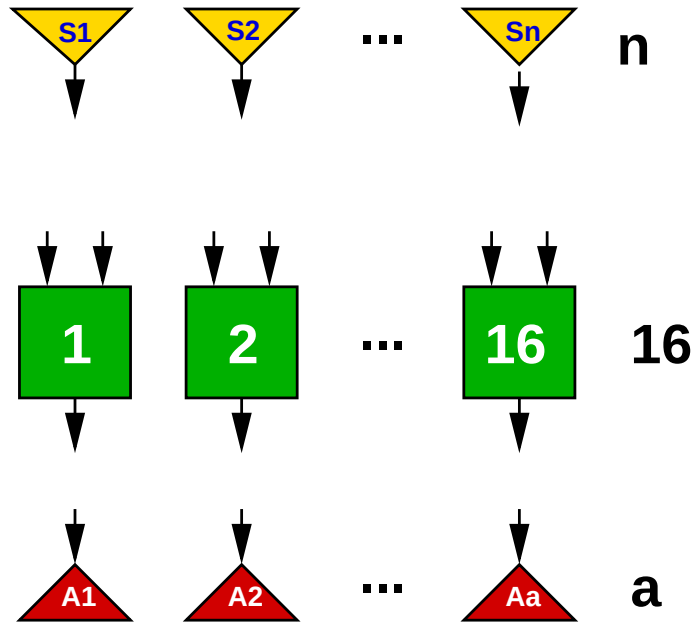
Ginsberg Critique (Ginsberg 89)

Universal plans are impossible to represent efficiently !



Ginsberg Critique (Ginsberg 89)

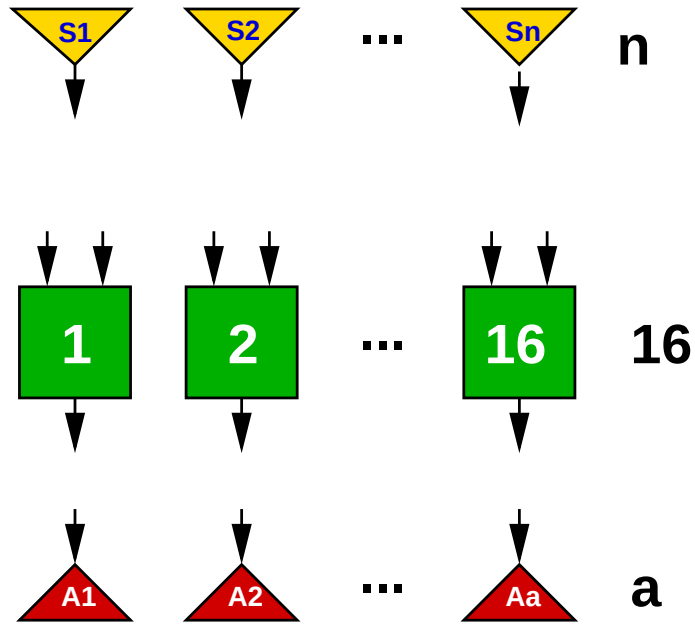
Universal plans are impossible to represent efficiently !



$(2^a)^{(2^n)}$ distinct universal plans

Ginsberg Critique (Ginsberg 89)

Universal plans are impossible to represent efficiently !



$(2^a)^{(2^n)}$ distinct universal plans

But at most $16^g(g+n)^{2g+a}$ distinct circuits of size g

Ginsberg's "Result"

- Not new: well known from Boolean function representation

Ginsberg's "Result"

- Not new: well known from Boolean function representation
- Problem: Ginsberg assumes a uniform distribution of important universal plans

Ginsberg's "Result"

- Not new: well known from Boolean function representation
- Problem: Ginsberg assumes a uniform distribution of important universal plans
- On the side: his "result" also applies to RL (but it only killed Universal Planning)

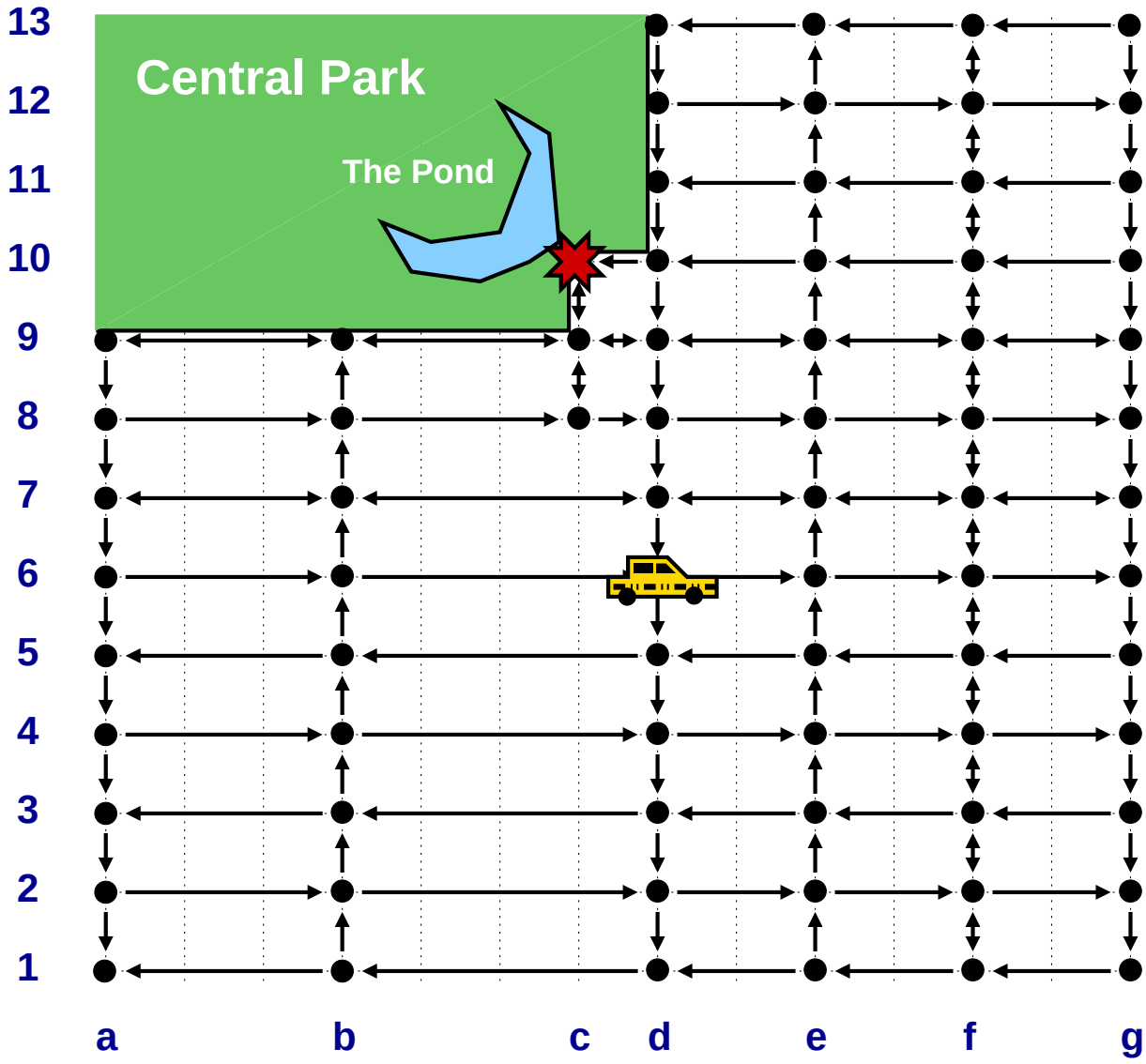
Universal Plan Representations

- Schoppers 87: Decision Trees

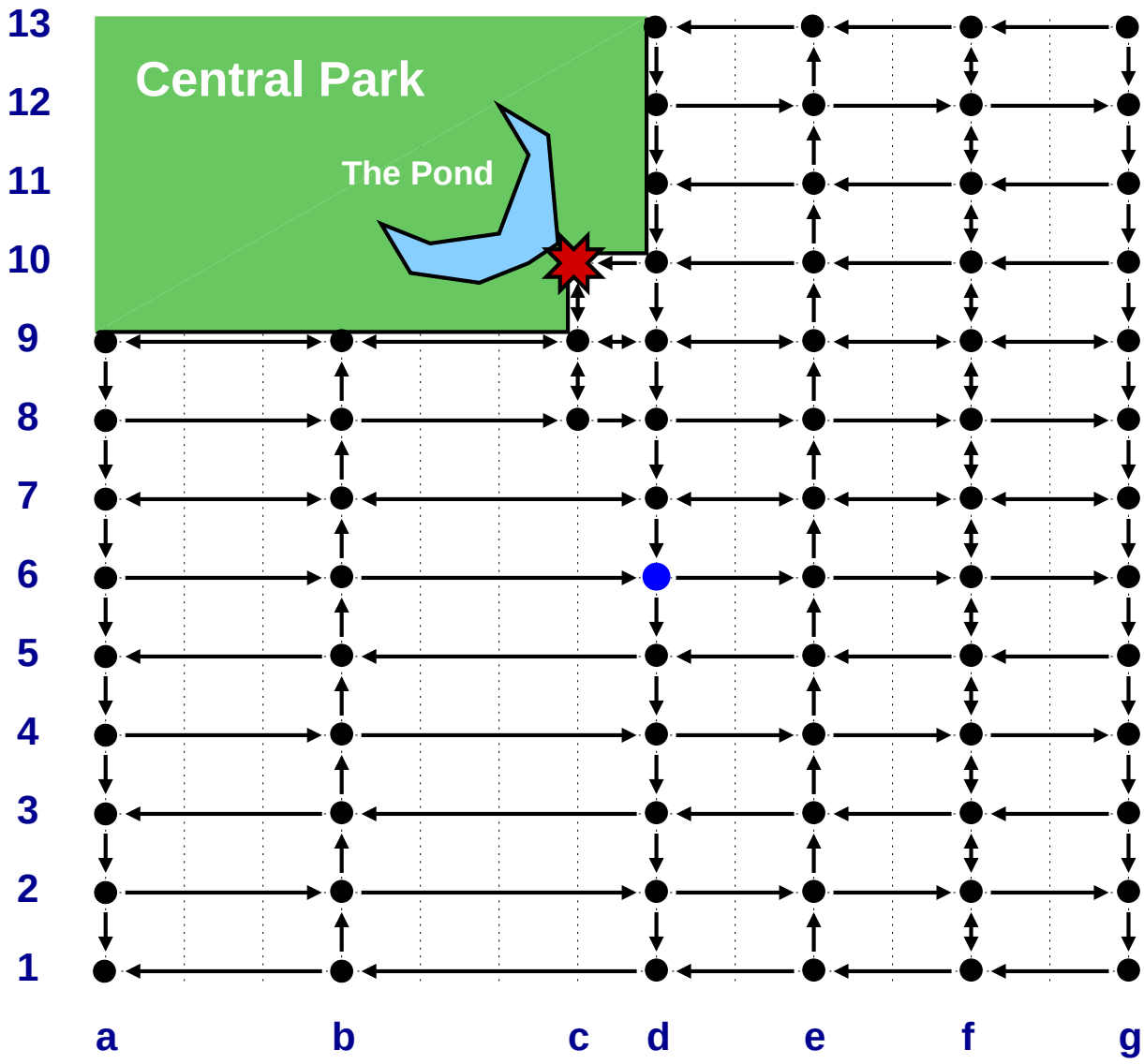
Universal Plan Representations

- Schoppers 87: Decision Trees
- Cimatti et al. 98: reduced Ordered Binary Decision Diagrams (OBDD,Bryant 86)

Usual Planning Algorithms

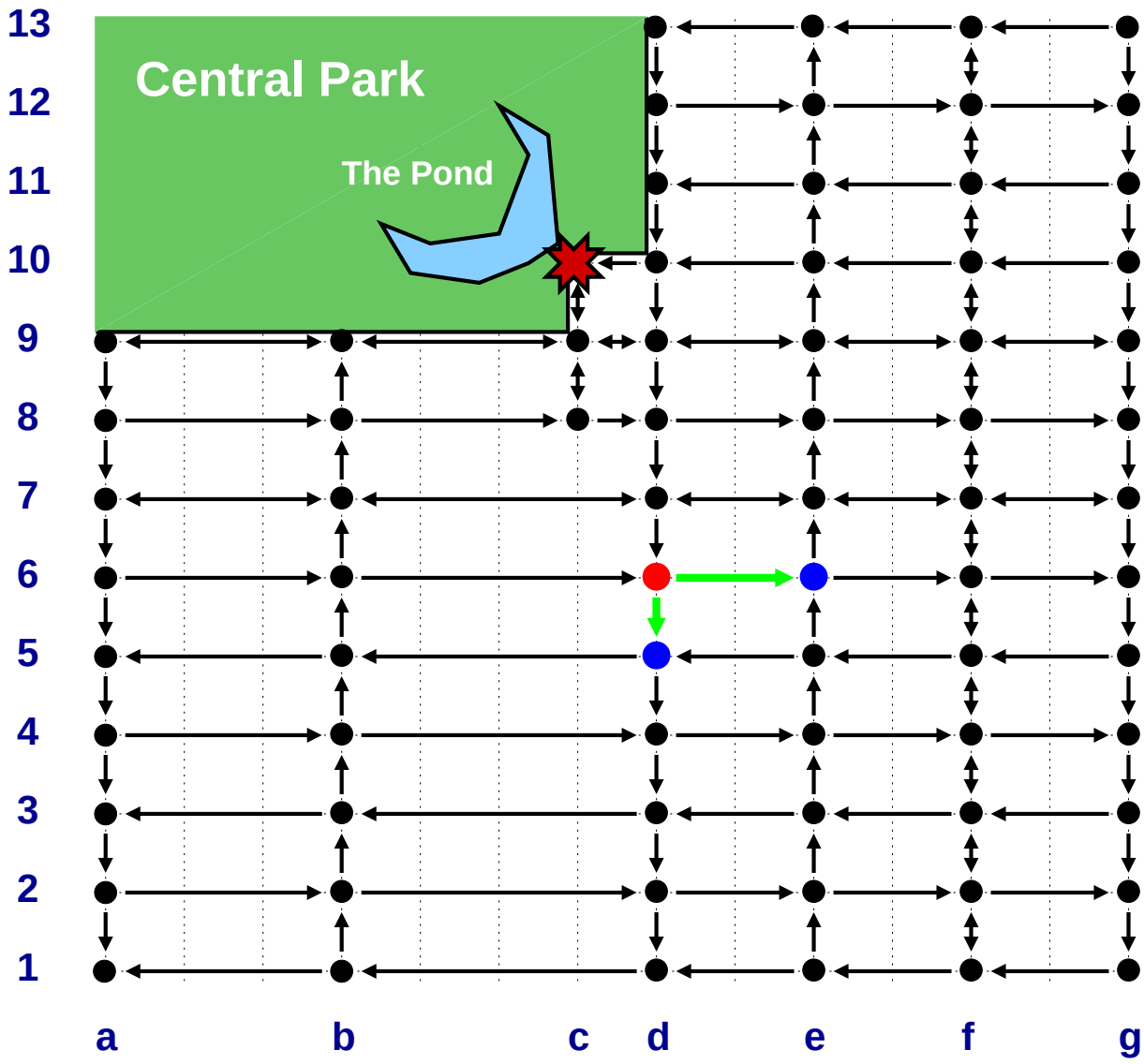


Usual Planning Algorithms



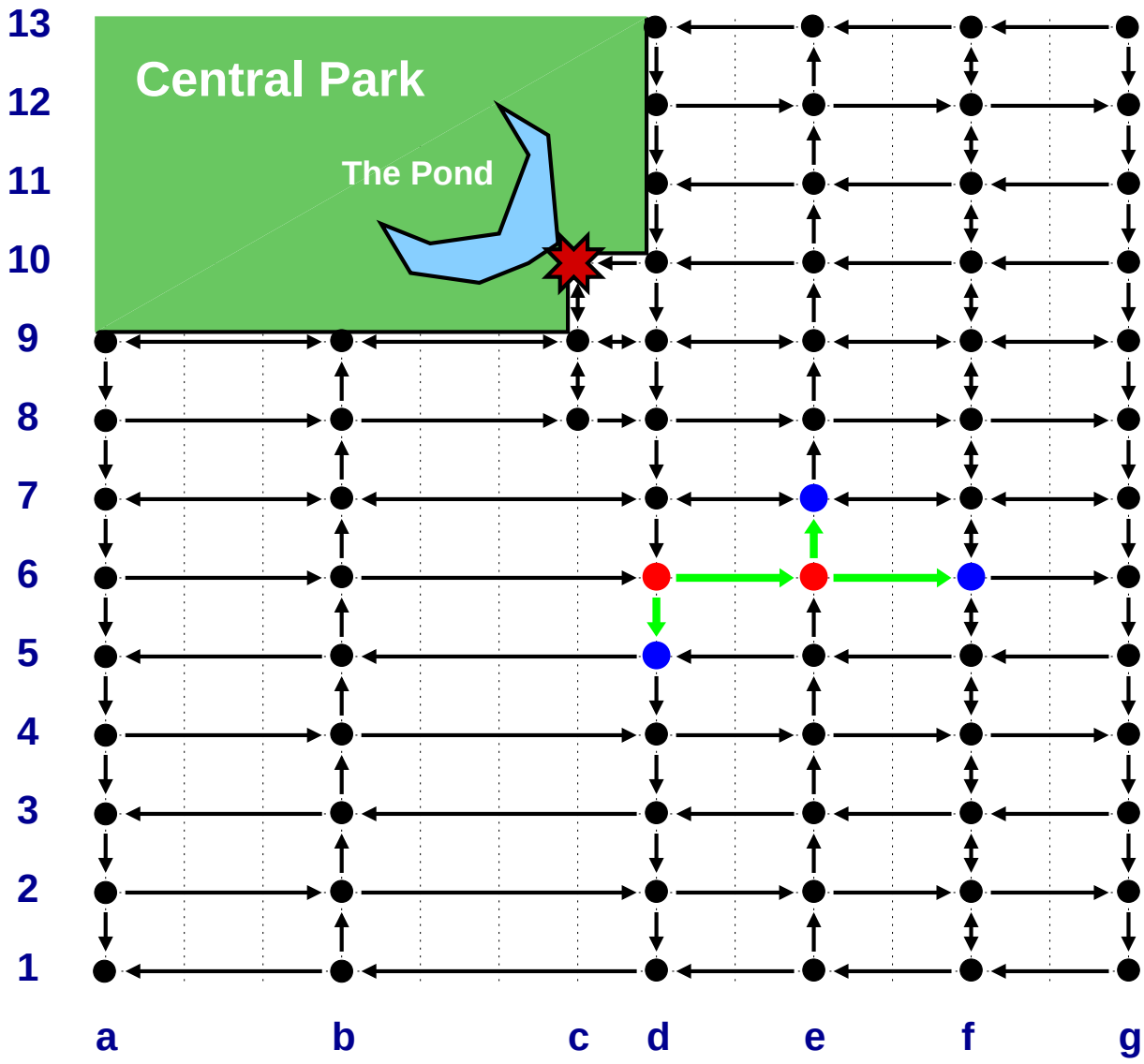
Expansion 0

Usual Planning Algorithms



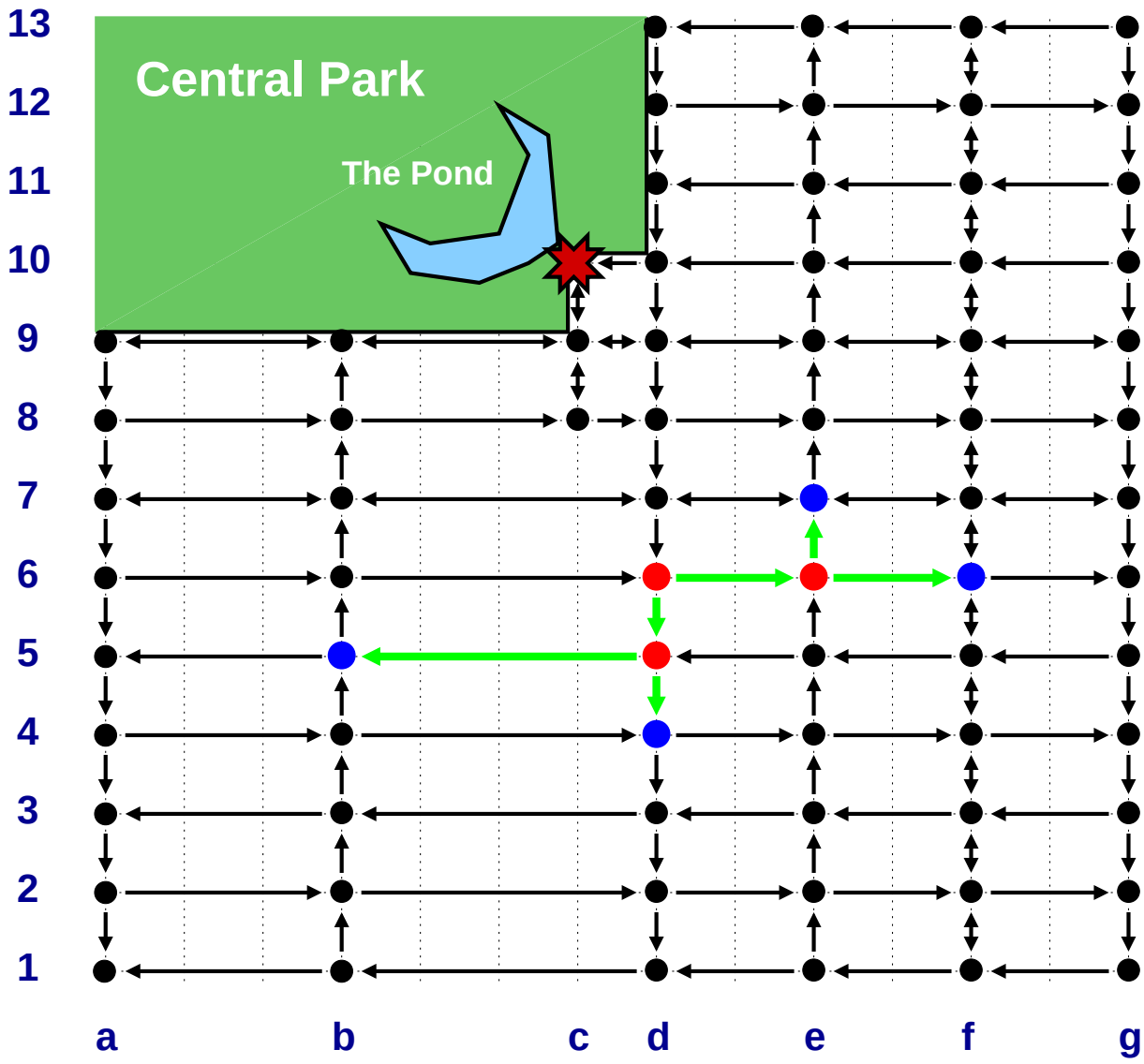
Expansion 1

Usual Planning Algorithms



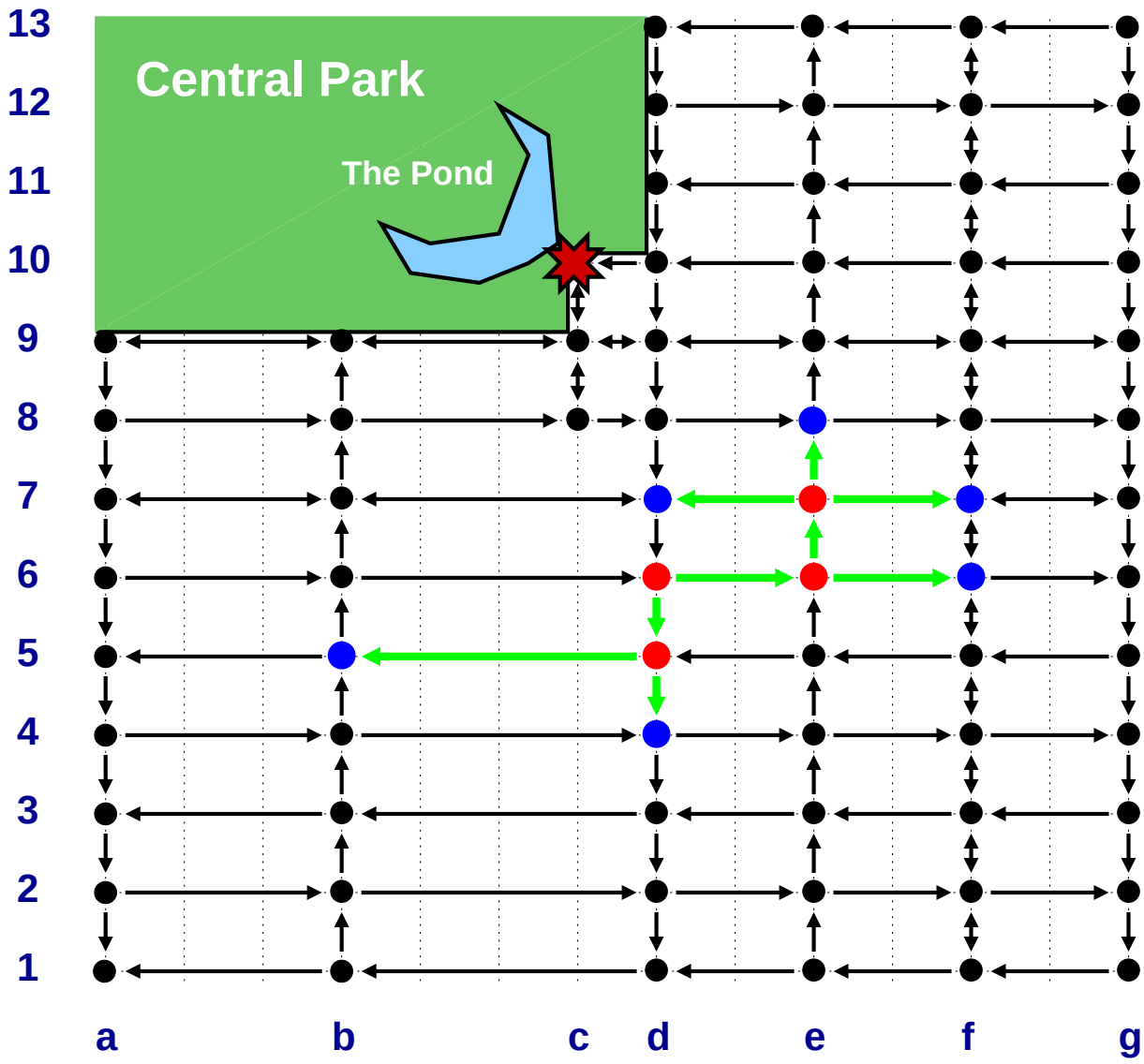
Expansion 2

Usual Planning Algorithms



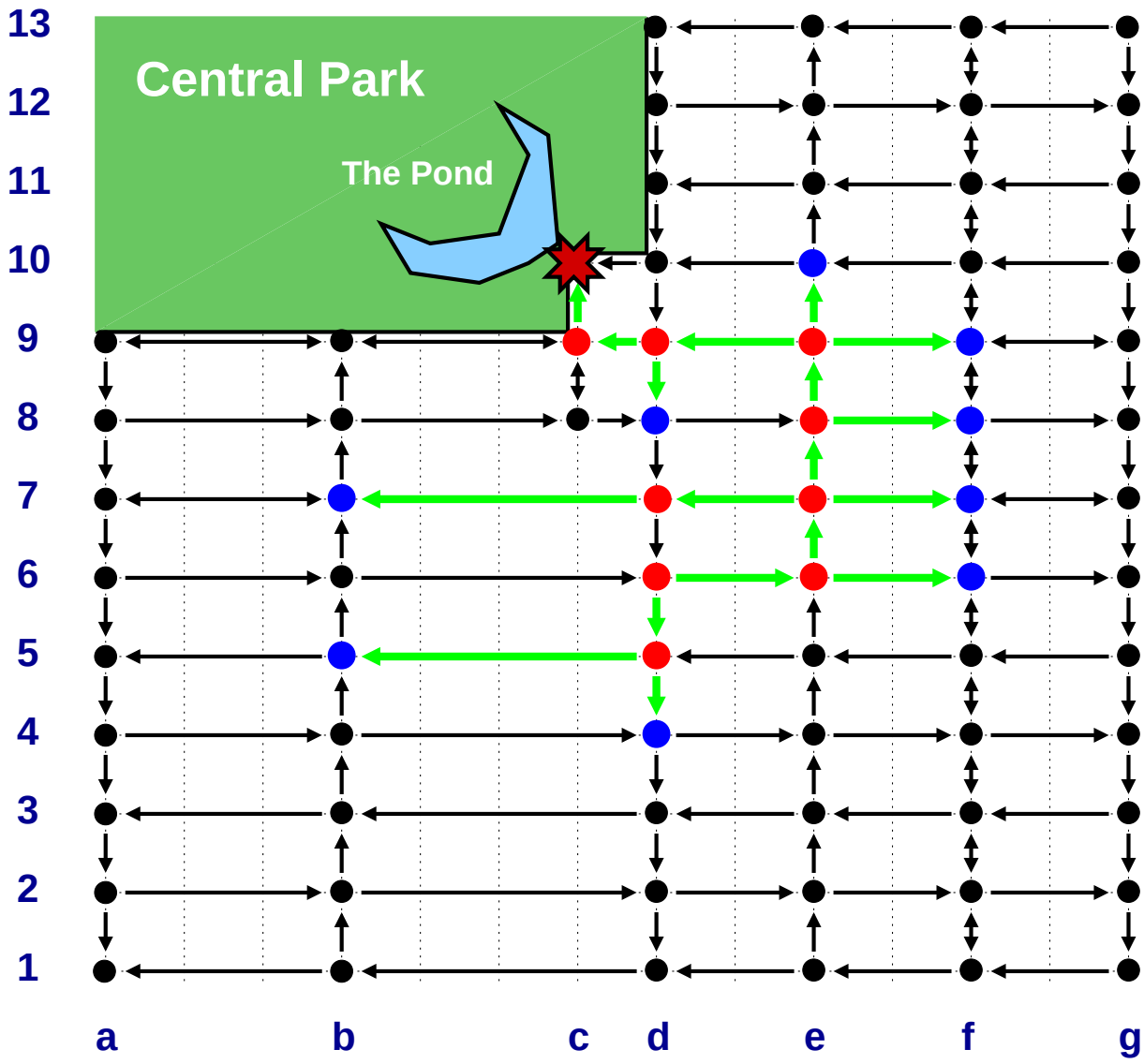
Expansion 3

Usual Planning Algorithms



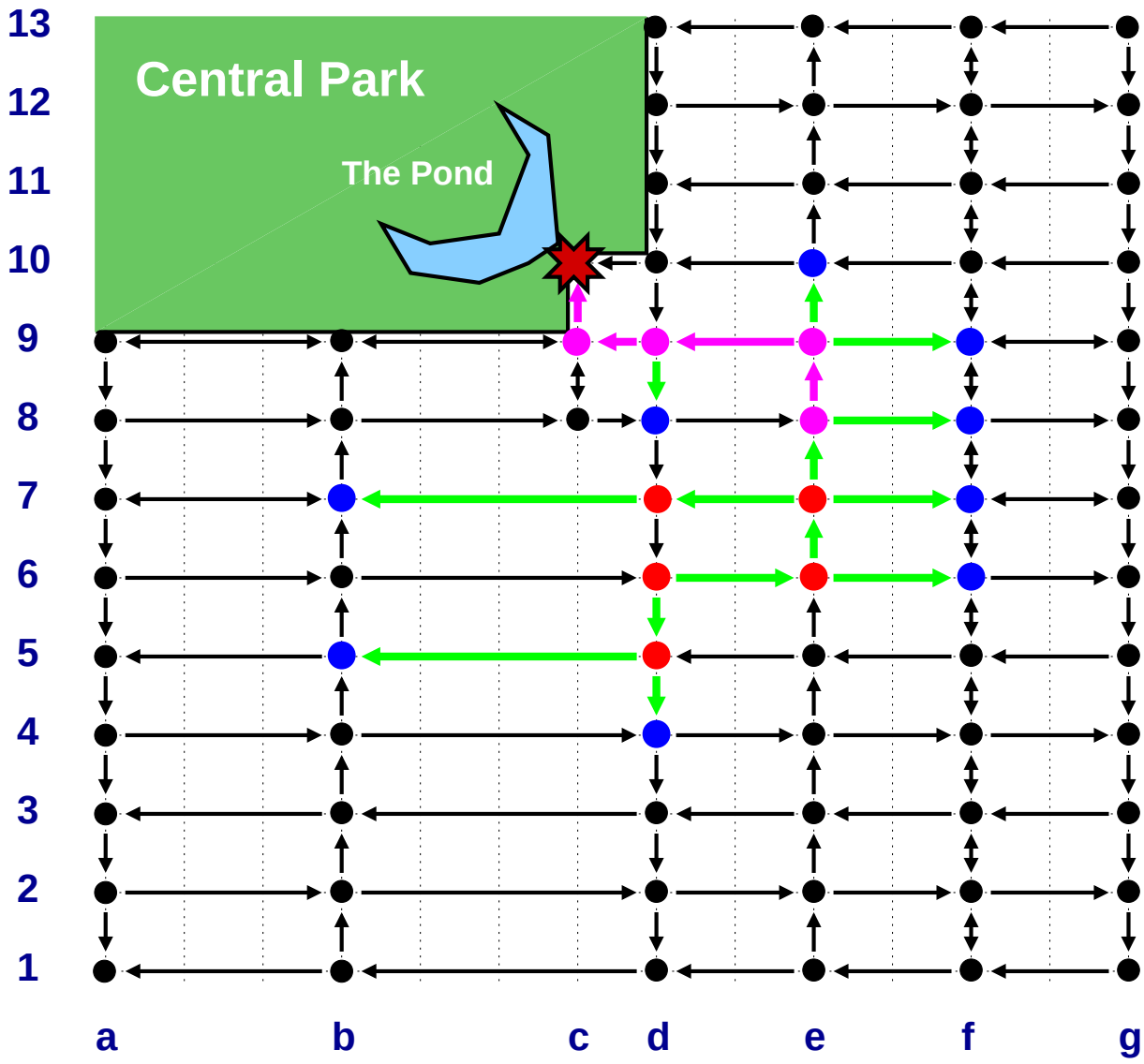
Expansion 4

Usual Planning Algorithms



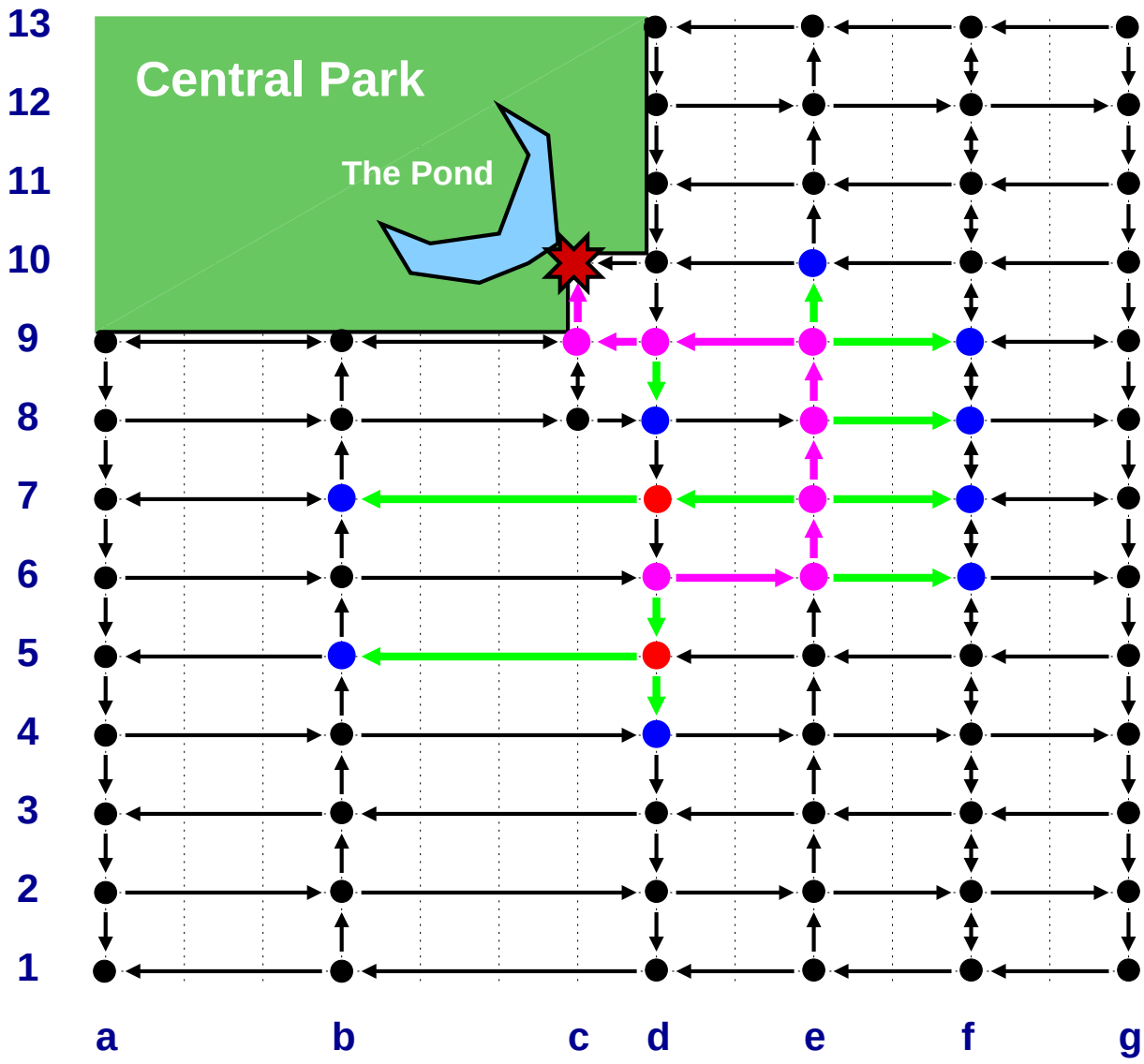
Expansion 9

Usual Planning Algorithms



Expansion 9

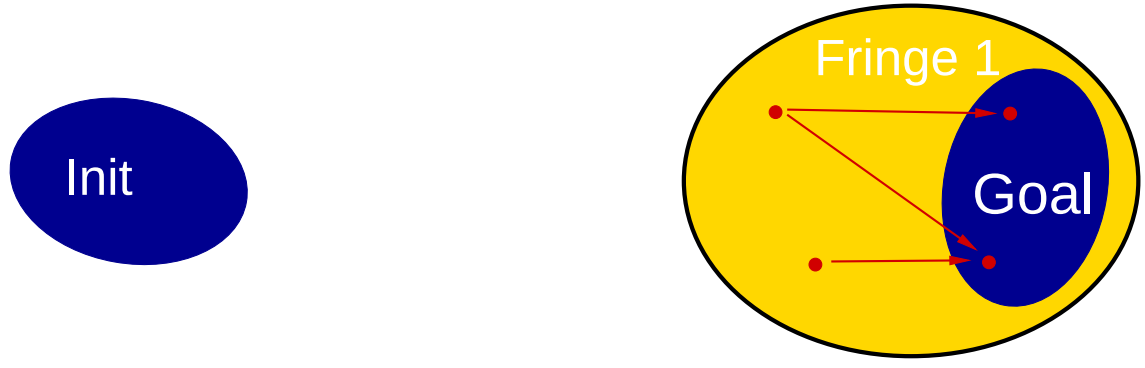
Usual Planning Algorithms



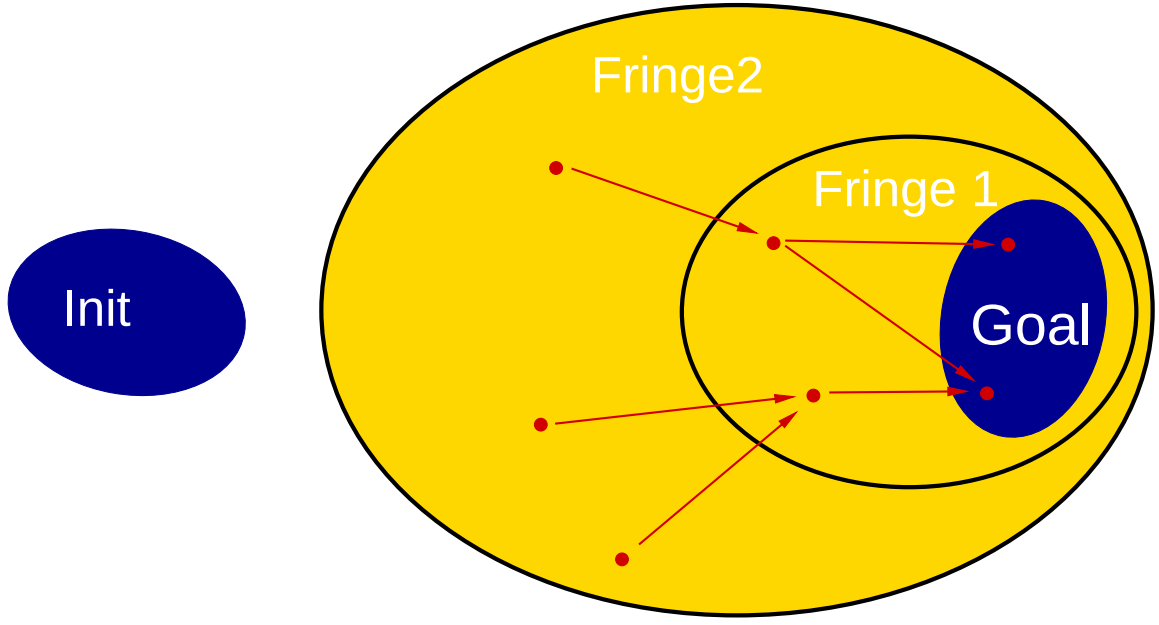
Expansion 9

Universal Planning Algorithms

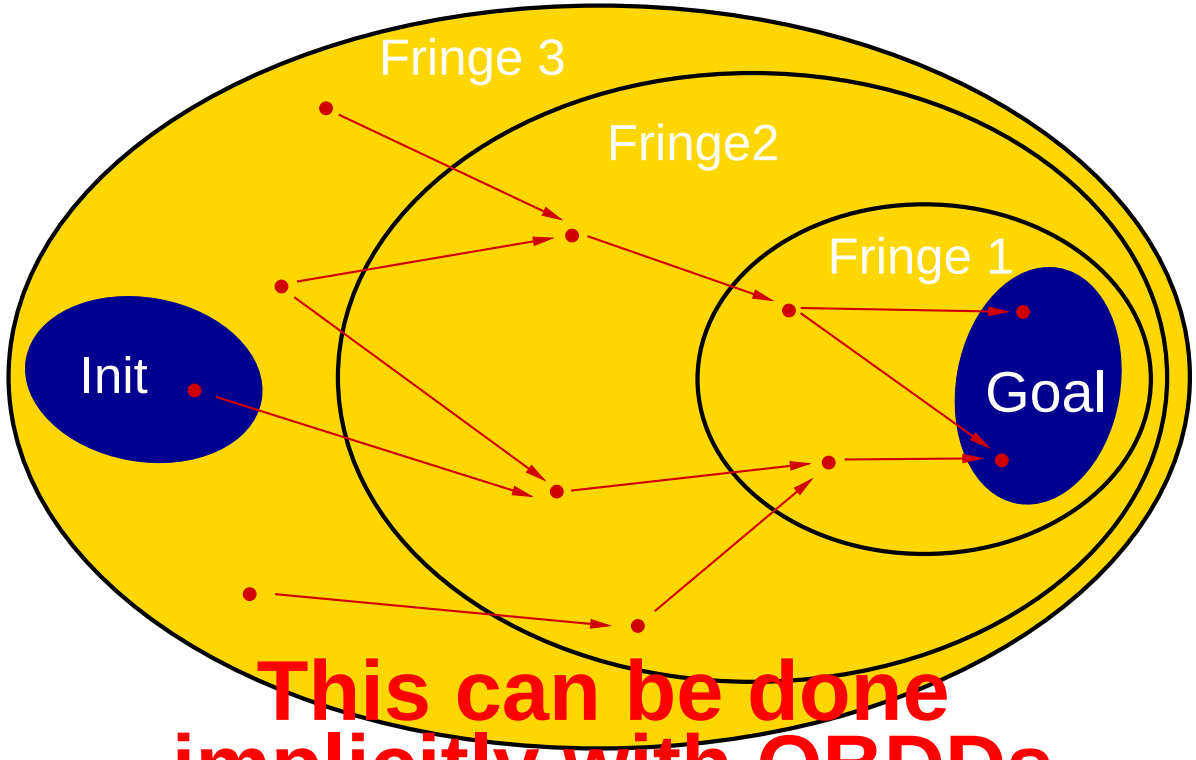
Universal Planning Algorithms



Universal Planning Algorithms



Universal Planning Algorithms



This can be done implicitly with OBDDs

The OBDD representation

An OBDD is a compact representation of a Boolean function

The OBDD representation

An OBDD is a compact representation of a Boolean function

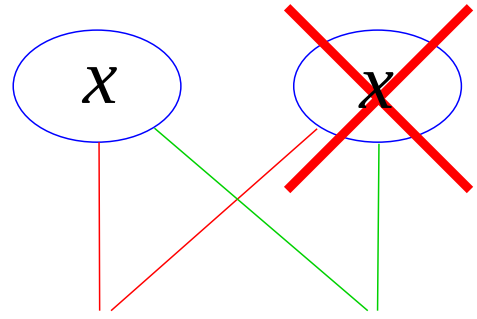
Definition

An OBDD is a DAG with,

- One or two terminal nodes labeled 0 or 1,
- A set of variable nodes with two edges low and $high$,
- A linear ordering of variables $x_1, x_2, \cdots, x_n$.

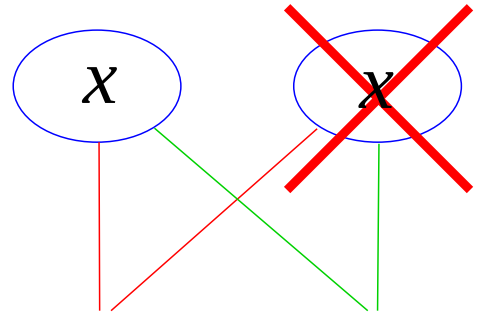
Reductions

1. Uniqueness of nodes associated to the same variable:

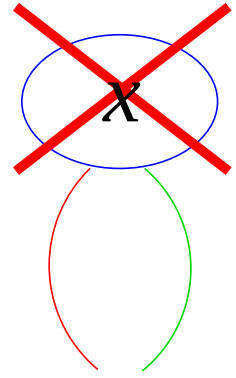


Reductions

1. Uniqueness of nodes associated to the same variable:

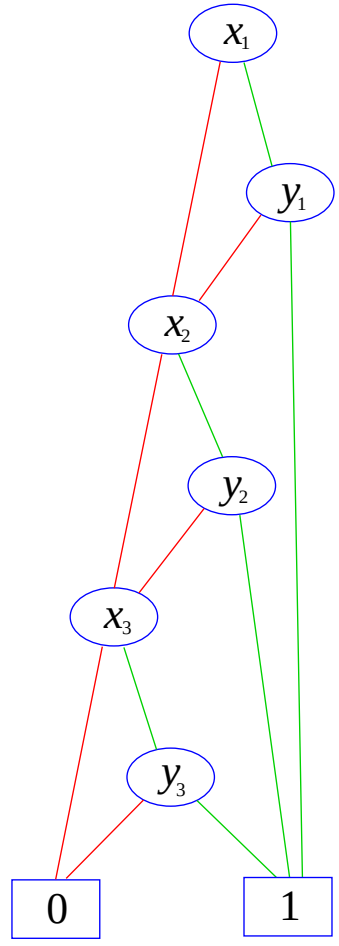


2. No redundant tests:



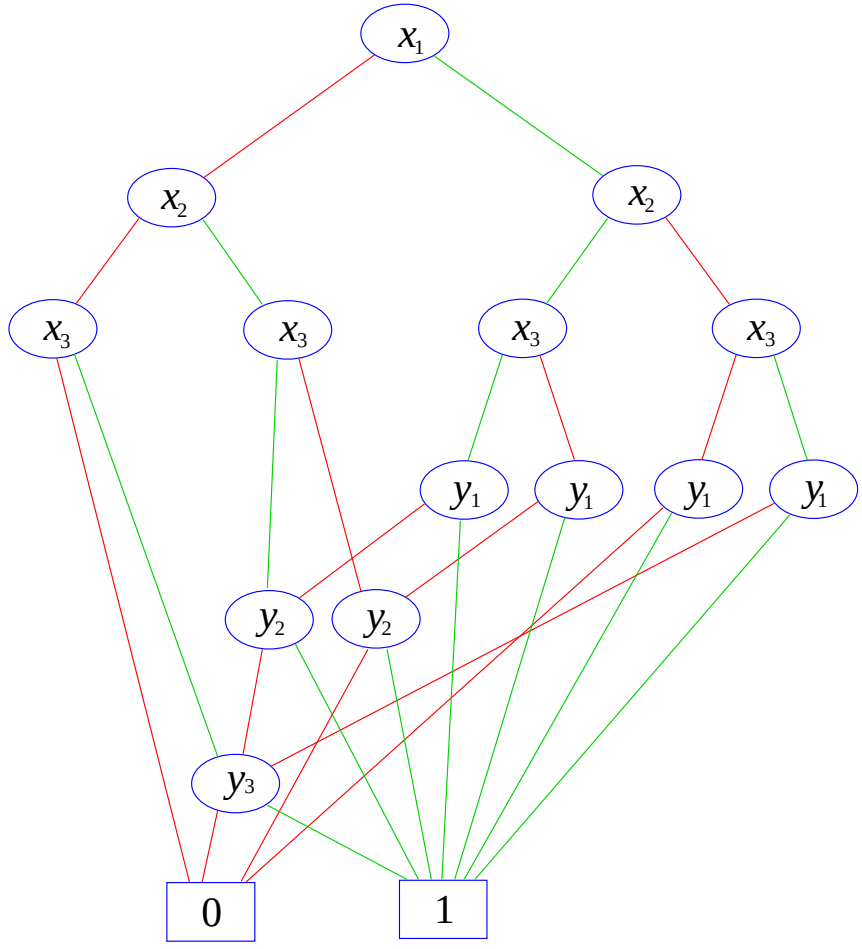
Result: A canonical representation !

A simple example



$$(x_1 \wedge y_1) \vee (x_2 \wedge y_2) \vee (x_3 \wedge y_3)$$

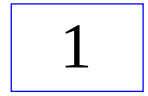
Exponential Blow Up
with bad Variable Ordering

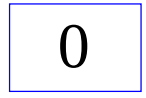


$$(x_1 \wedge y_1) \vee (x_2 \wedge y_2) \vee (x_3 \wedge y_3)$$

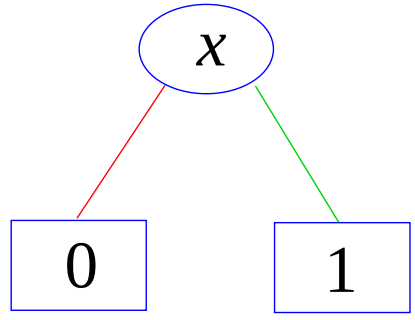
Building Expressions: base cases

- **True, False** $O(1)$

$true$ 

$false$ 

- **Variables** $O(1)$

x 

Building Expressions: inductive cases

- **Binary operations** $O(|f||g|)$:

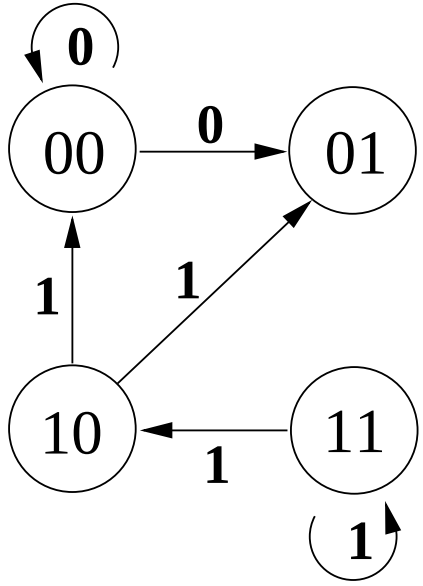
$$
\begin{aligned}
& f \oplus g \\
= \quad & (x_1 \rightarrow f_{high}, f_{low}) \oplus (x_1 \rightarrow g_{high}, g_{low}) \\
= \quad & x_1 \rightarrow (f_{high} \oplus g_{high}), (f_{low} \oplus g_{low}) \\
= \quad & \cdots
\end{aligned}
$$

- **Unary operations** $O(|f|)$:

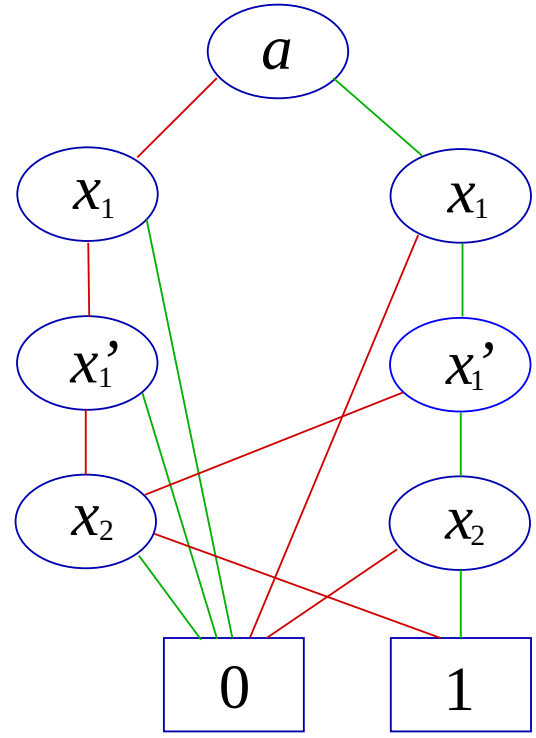
  Implemented with binary operations

Non-deterministic Domains represented by OBDDs

FSM

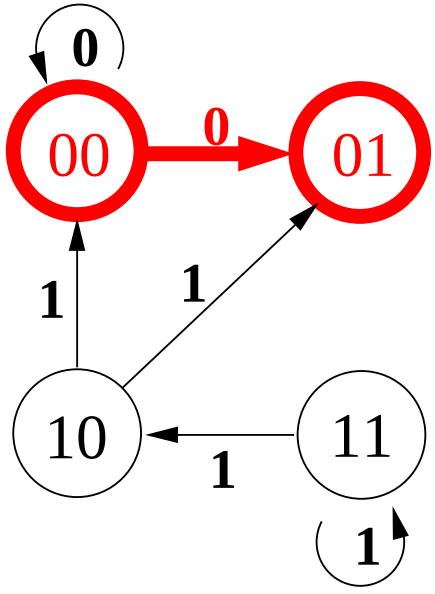


$T(a,x_1,x_1',x_2,x_2')$

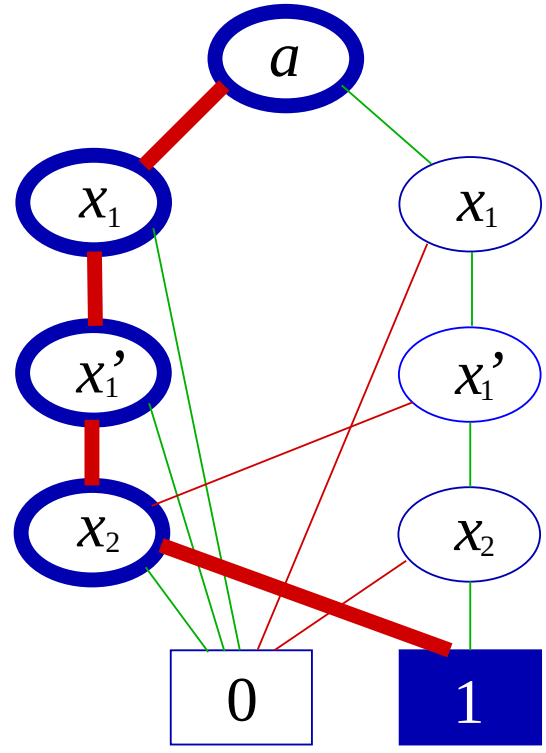


Non-deterministic Domains represented by OBDDs

FSM



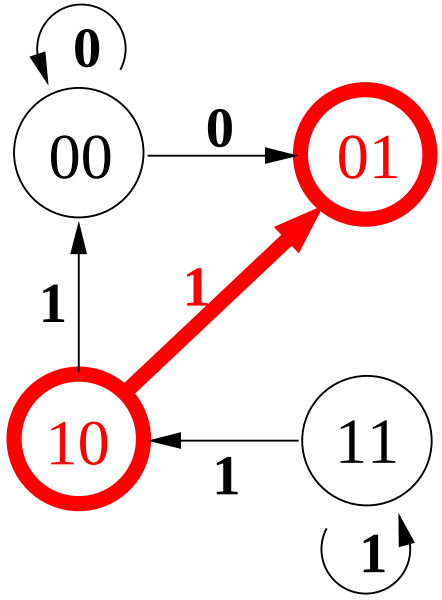
$T(a,x_1,x_1',x_2,x_2')$



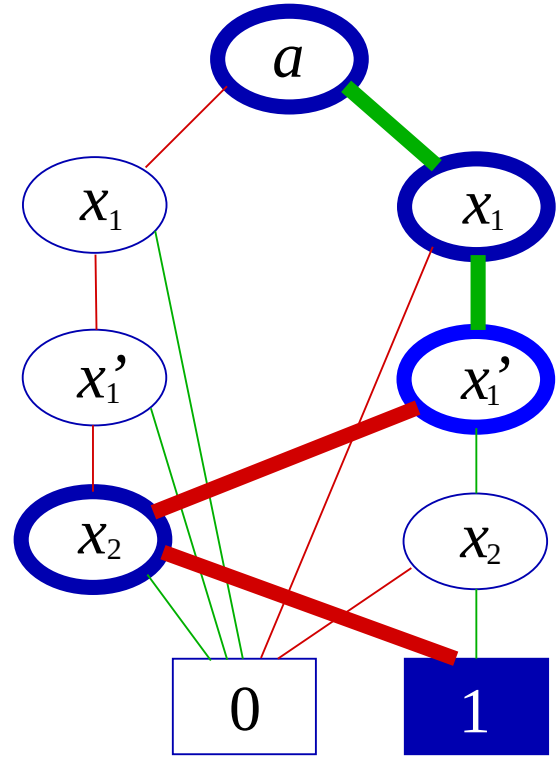
T(0,0,0,0,1) = 1

Non-deterministic Domains represented by OBDDs

FSM



$T(a,x_1,x_1',x_2,x_2')$



T(1,1,0,0,1) = 1

OBDD-based Planning

Action 0

Pre: $\neg x_1 \wedge \neg x_2$

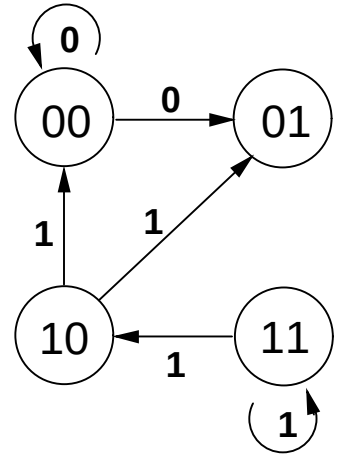
Eff: $\neg x_1'$

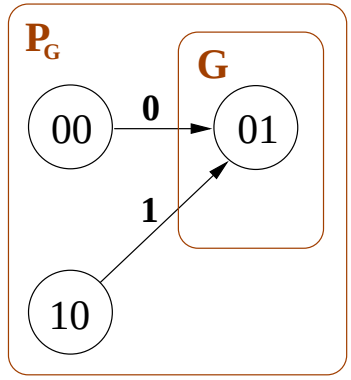
Action 1

Pre: x_1

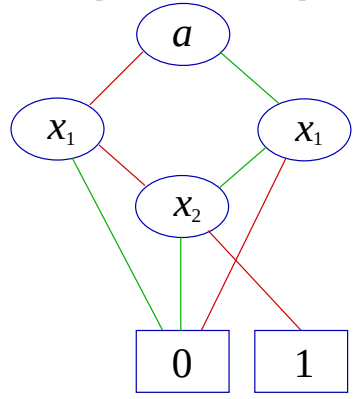
Eff: $(x_1 \wedge \neg x_2 \rightarrow \neg x_1') \wedge (x_1 \wedge x_2 \rightarrow x_1')$

FSM





$P_G(a,x_1,x_2)$



$$\mathrm{P_V}(a, x_1, x_2) = \exists x_1', x_2' \,.\, \mathrm{T}(a, x_1, x_1', x_2, x_2') \wedge \mathrm{V}(x_1', x_2')$$

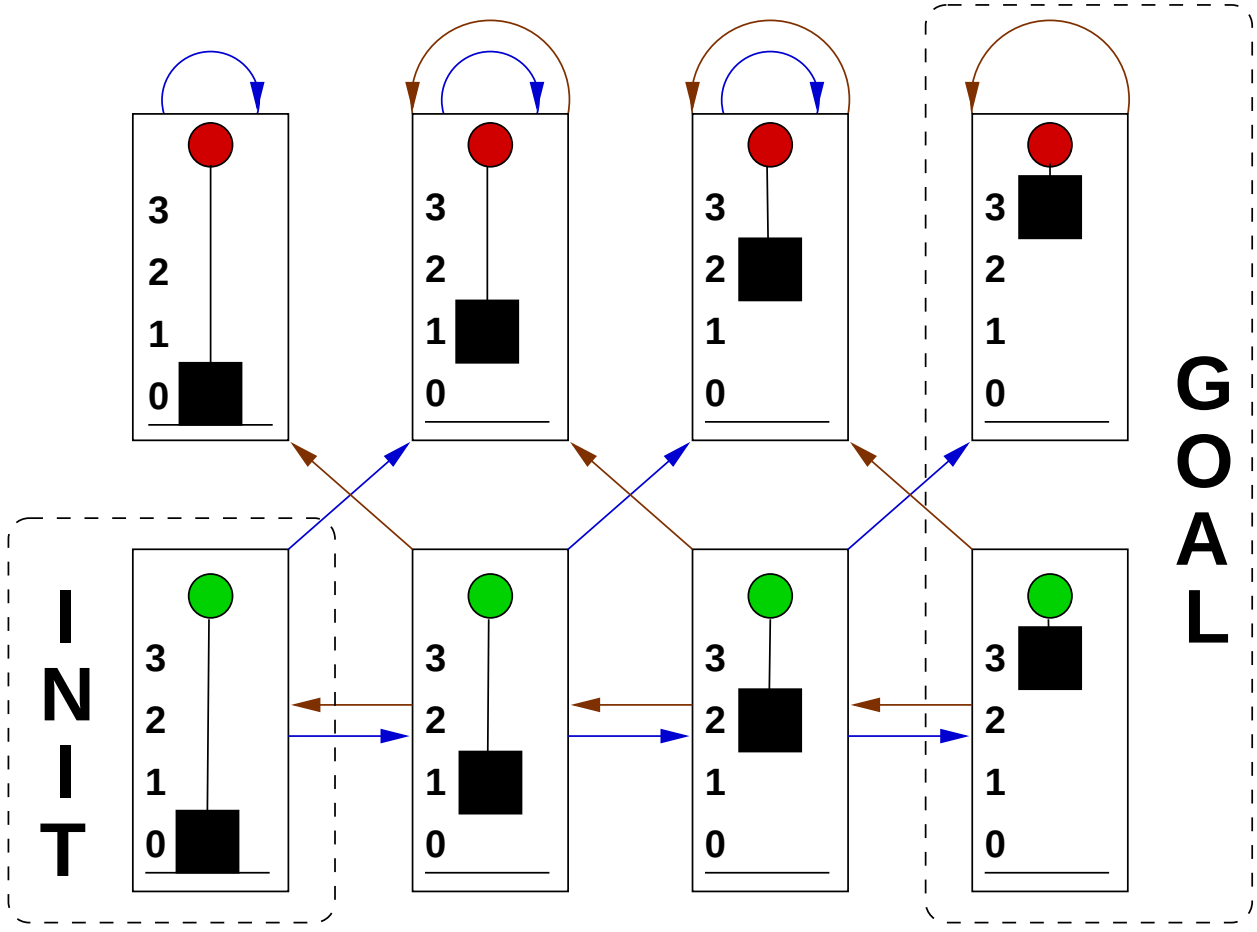
$$\mathrm{P_G}(a, x_1, x_2) = \exists x_1', x_2' \,.\, \mathrm{T}(a, x_1, x_1', x_2, x_2') \wedge (\neg x_1' \wedge x_2')$$

The NADL Domain Language

An NADL domain description consists of:

- A set of propositional and numerical state variables
- Two sets of agents:
  - System agents (at least one)
  - Environment agents (maybe none)
- A nonempty set of actions associated to each agent
- A set of initial and a goal states

Example



NADL Description

variables

nat(4) pos **bool** $lift\_works$

system

agt: Lift

Up

mod: pos

pre: $pos < 3$

eff: $lift\_works \rightarrow pos' = pos + 1, pos' = pos$

Down

mod: pos

pre: $pos > 0$

eff: $lift\_works \rightarrow pos' = pos - 1, pos' = pos$

environment

agt: Env

Break

mod: $lift\_works$

pre: $true$

eff: $\neg lift\_works \Rightarrow \neg lift\_works'$

initially

$pos = 0 \wedge lift\_works$

goal

$pos = 3$

OBDD Encoding of NADL Domains

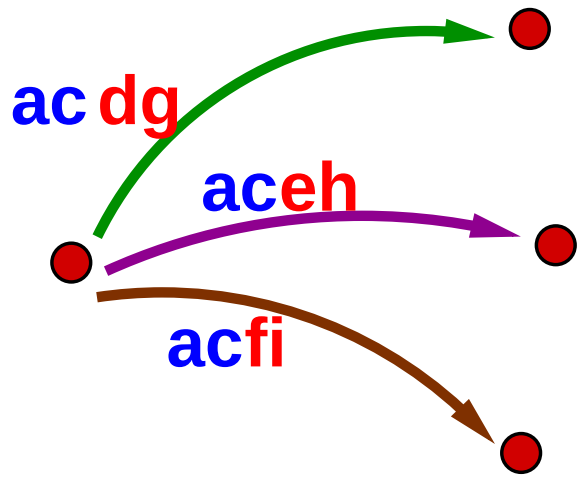
- Use an OBDD to represent all legal joint actions of environment and system agents

OBDD Encoding of NADL Domains

- Use an OBDD to represent all legal joint actions of environment and system agents
- Remove action labels of environment actions

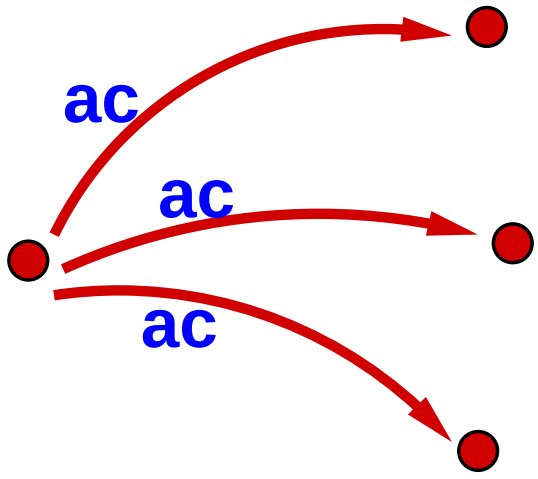
OBDD Encoding of NADL Domains

- Use an OBDD to represent all legal joint actions of environment and system agents
- Remove action labels of environment actions



OBDD Encoding of NADL Domains

- Use an OBDD to represent all legal joint actions of environment and system agents
- Remove action labels of environment actions



OBDD-based Universal Planning

function *Plan*(T, I, G)

$U := \emptyset; V := G$

while $I \not\subset V$

$U_c := PreComp(T, V)$

if $U_c = \emptyset$ **then**

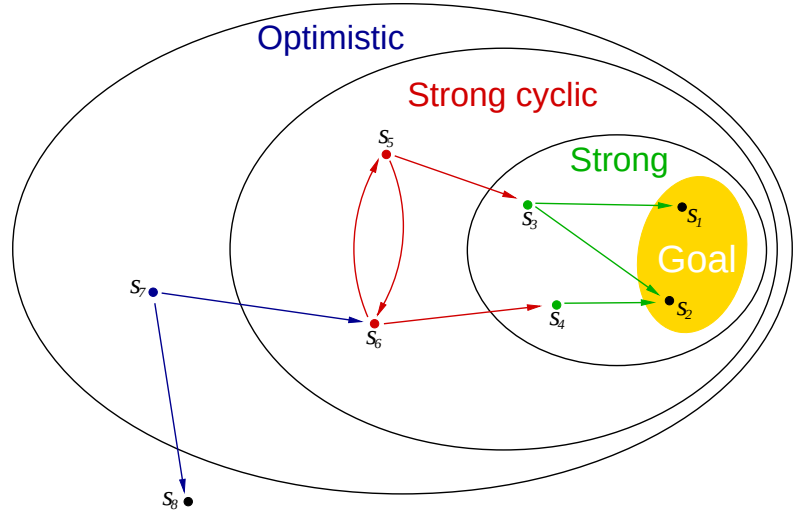
return *failure*

else

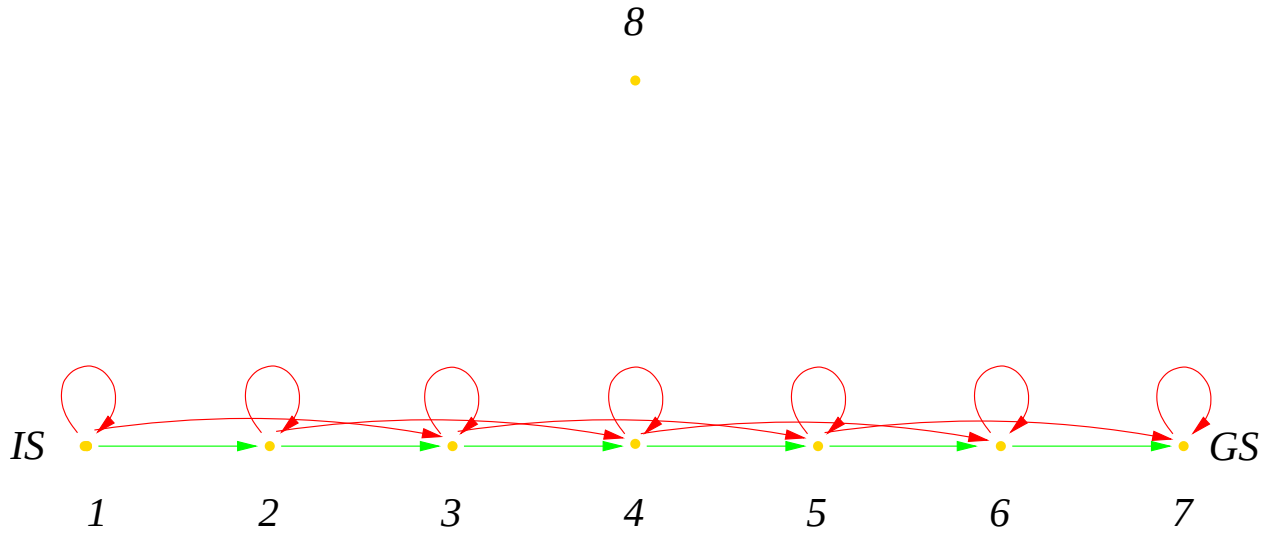
$U := U \cup U_c$

$V := V \cup states(U_c)$

return U

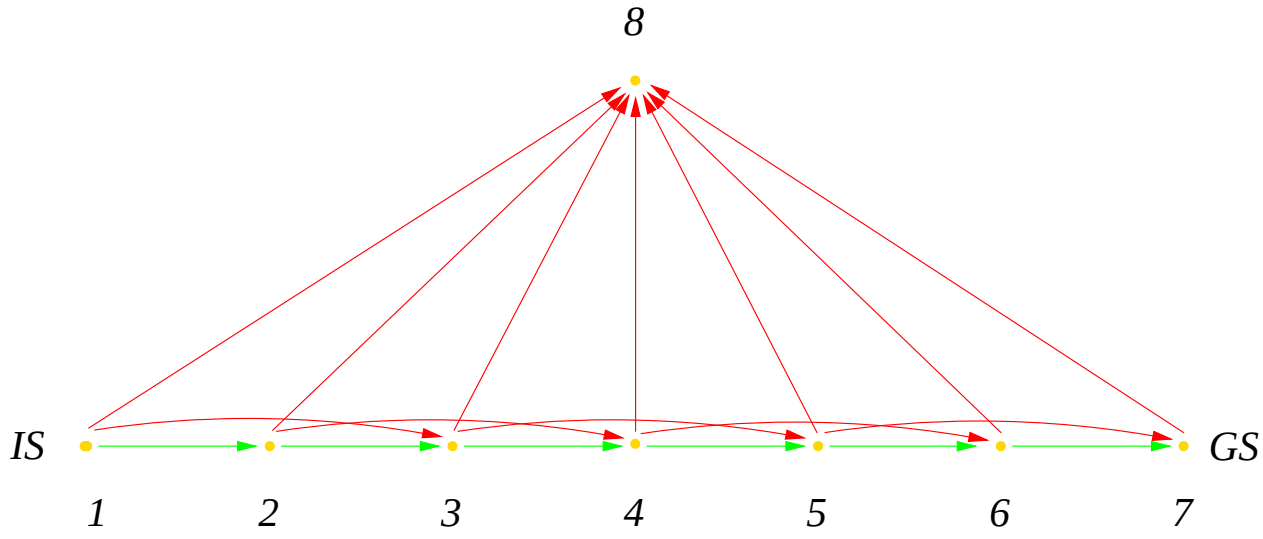


Optimistic and StrongCyclic I



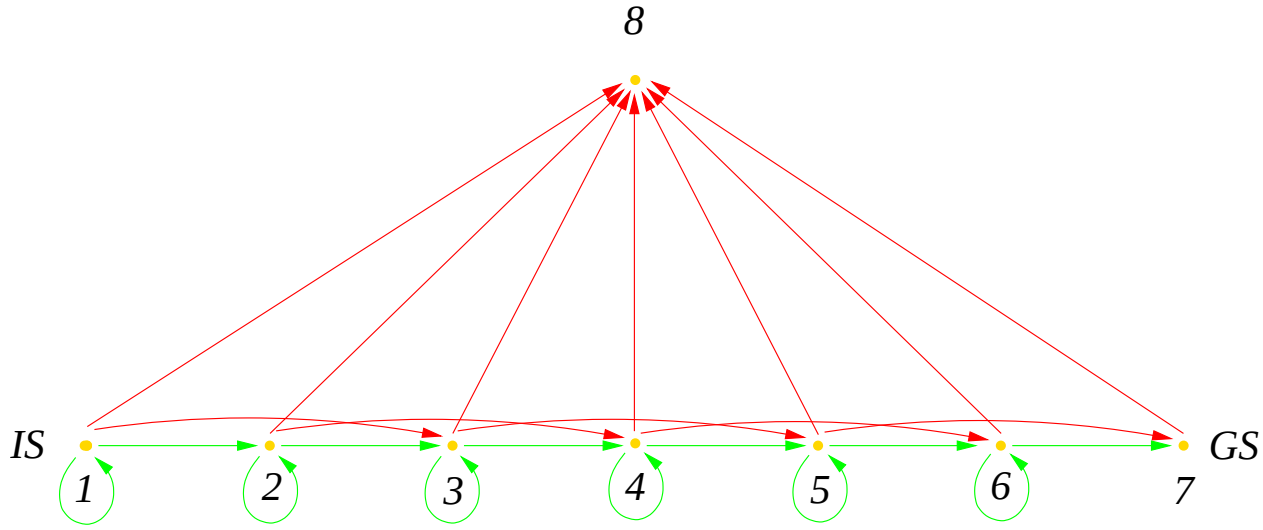
- StrongCyclic (= strong): Best: 6 Worst: 6
- Optimistic: Best: 3 Worst: ∞

Optimistic and StrongCyclic II



- StrongCyclic (= strong): Best: 6 Worst: 6
- Optimistic: Best: 3 Worst: Dead End

Optimistic and StrongCyclic III



- StrongCyclic: Best: 6 Worst: ∞
- Optimistic: Best: 3 Worst: Dead End

UMOP (Jensen & Veloso 99)

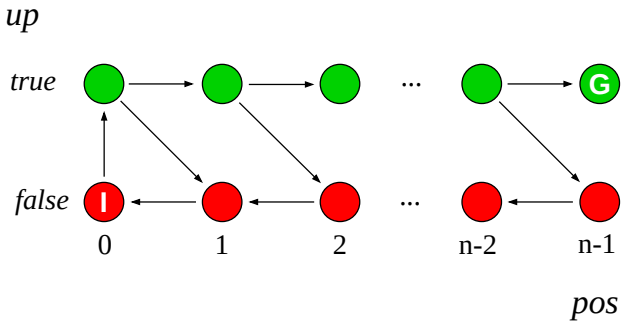
- Input: NADL ASCII file
- Output: OBDD file of universal plan
- Algorithms:
  - Strong planning
  - Strong cyclic planning
  - Optimistic planning

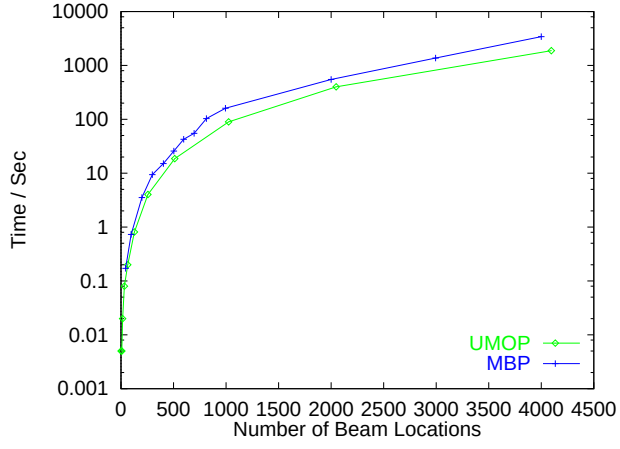
Get version 1.2 from:

`www.cs.cmu.edu/~runej/publications/umop.html`

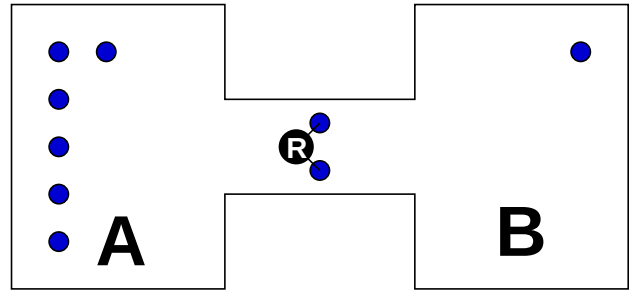
Comparison Experiments

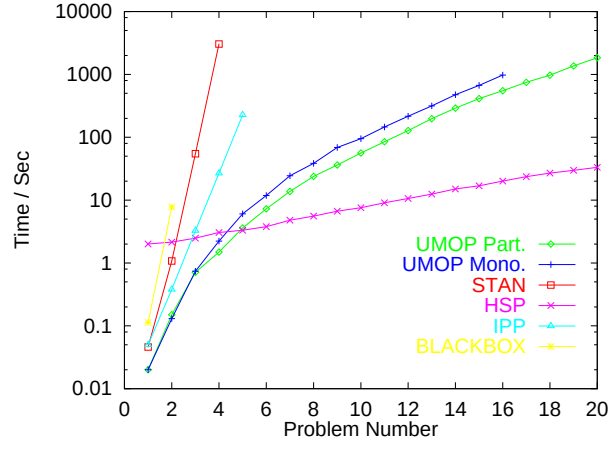
Beam Walk (MBP)





Gripper (AIPS'98)

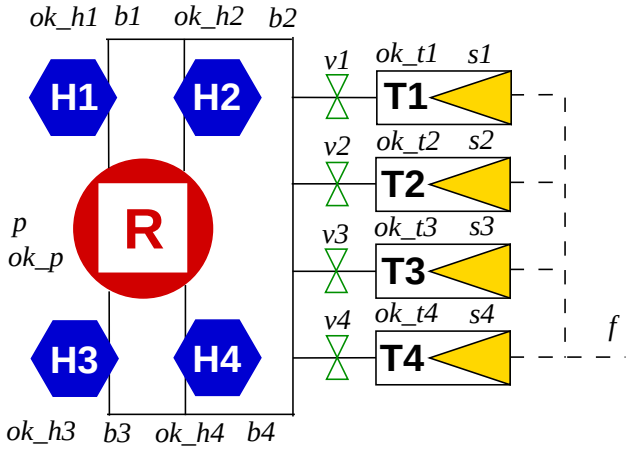




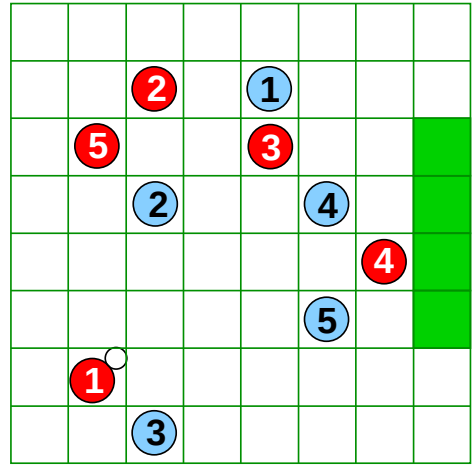
UMOP mono. = MBP ! Partitioning scales up 2x !

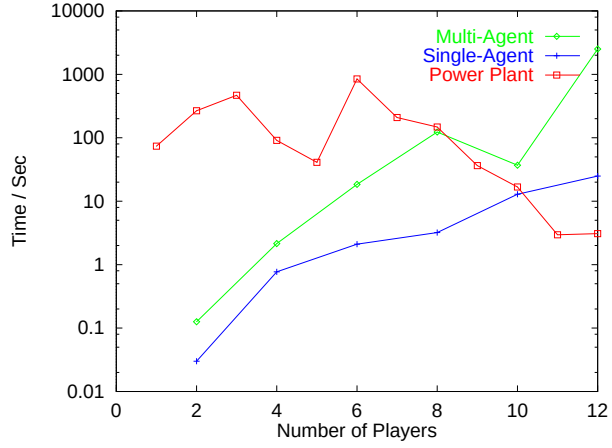
Multi-Agent Domains

Power plant



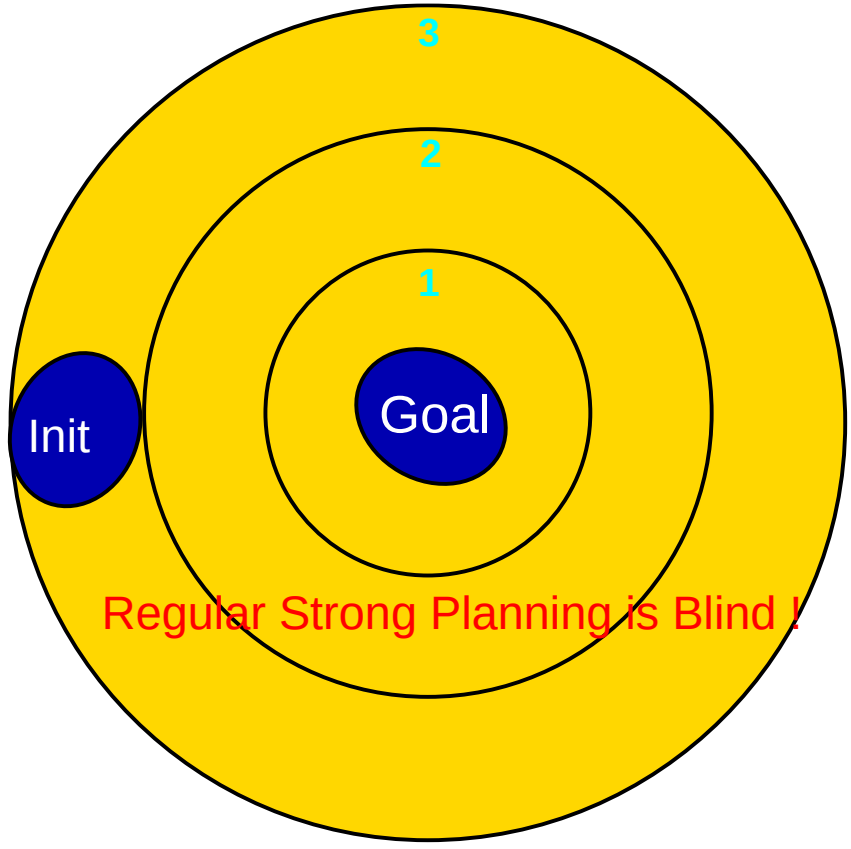
Soccer



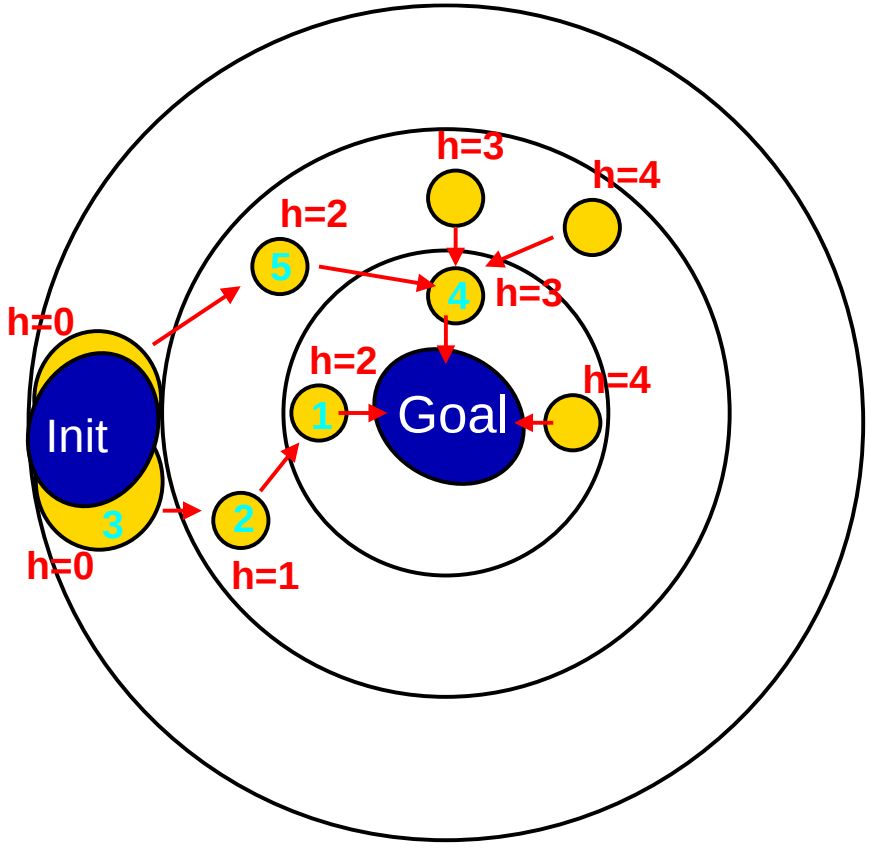


More agents might be better !

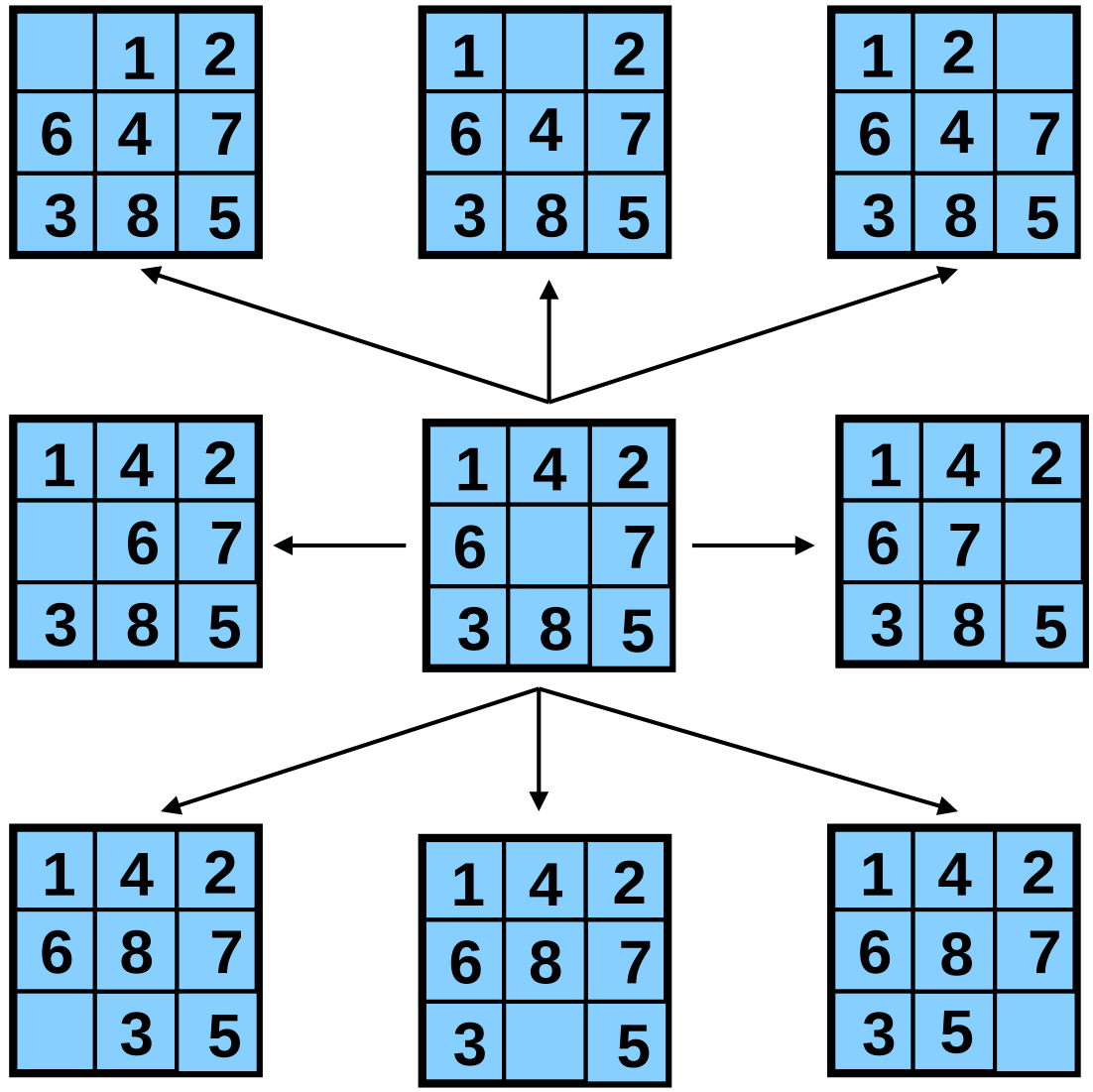
Guided Strong Planning



Guided Strong Planning



Non-Det. 8-Puzzle



Results

	Strong	Hstrong
it	24	316
$\|fringe\|$	6638	49
$\|sol\|$	73461	4159
T_{total}	37.64	8.32
T_{search}	34.94	6.05

Other Recent Work

- OBDD-based universal planning in adversarial domains (Jensen, Veloso, & Bowling 01)
- OBDD-based planning in belief space (Bertoli et al. 01)
- Universal OBDD-based planning for extended goals (Pistore et al. 01)
- (Efficient) Heuristic OBDD-based deterministic planning (Jensen, Veloso, & Bryant 02)
- 1-fault tolerant OBDD-based planning (Jensen, Veloso, & Bryant 02?)