State Abstraction Discovery from Irrelevant State Variables

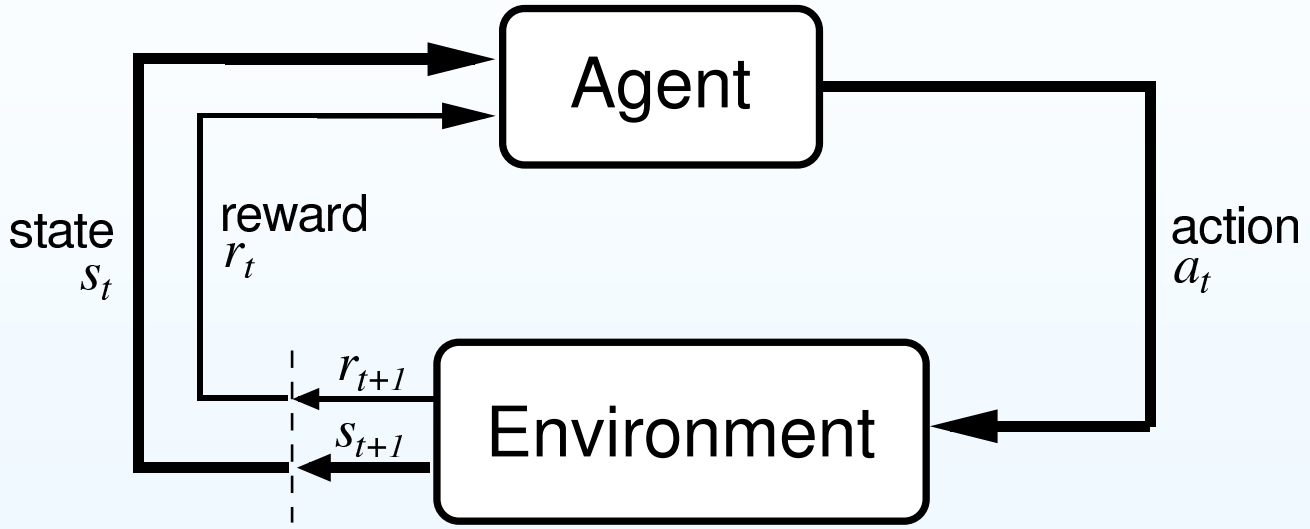
Nicholas K. Jong and Peter Stone

{nkj,pstone}@cs.utexas.edu.

Department of Computer Sciences

The University of Texas at Austin

Reinforcement Learning



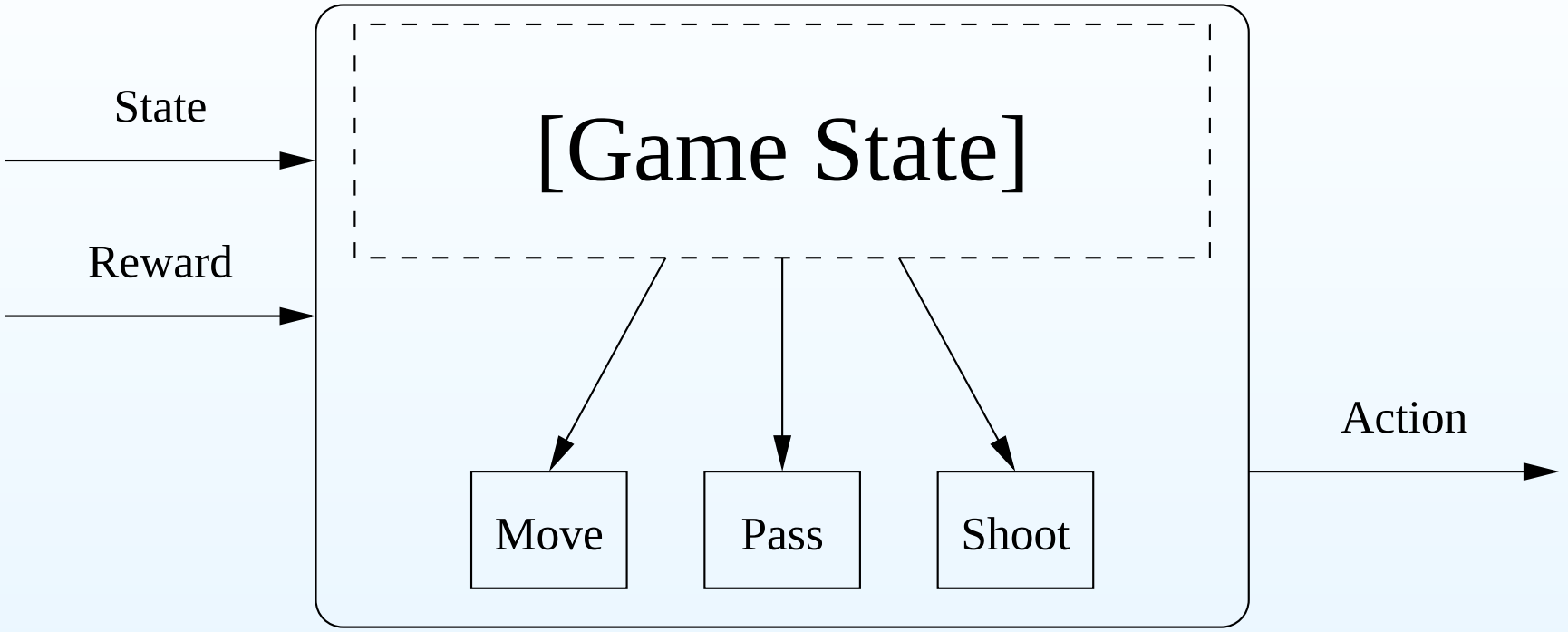
- Task: Maximize rewards in an unknown environment
- Only given: the state-action interface
- Much research: learn policies given an arbitrary interfaces
- Our research: discover interfaces that are easier to learn

Value-Based RL

Learn: a control policy

“What action should I choose in each state?”

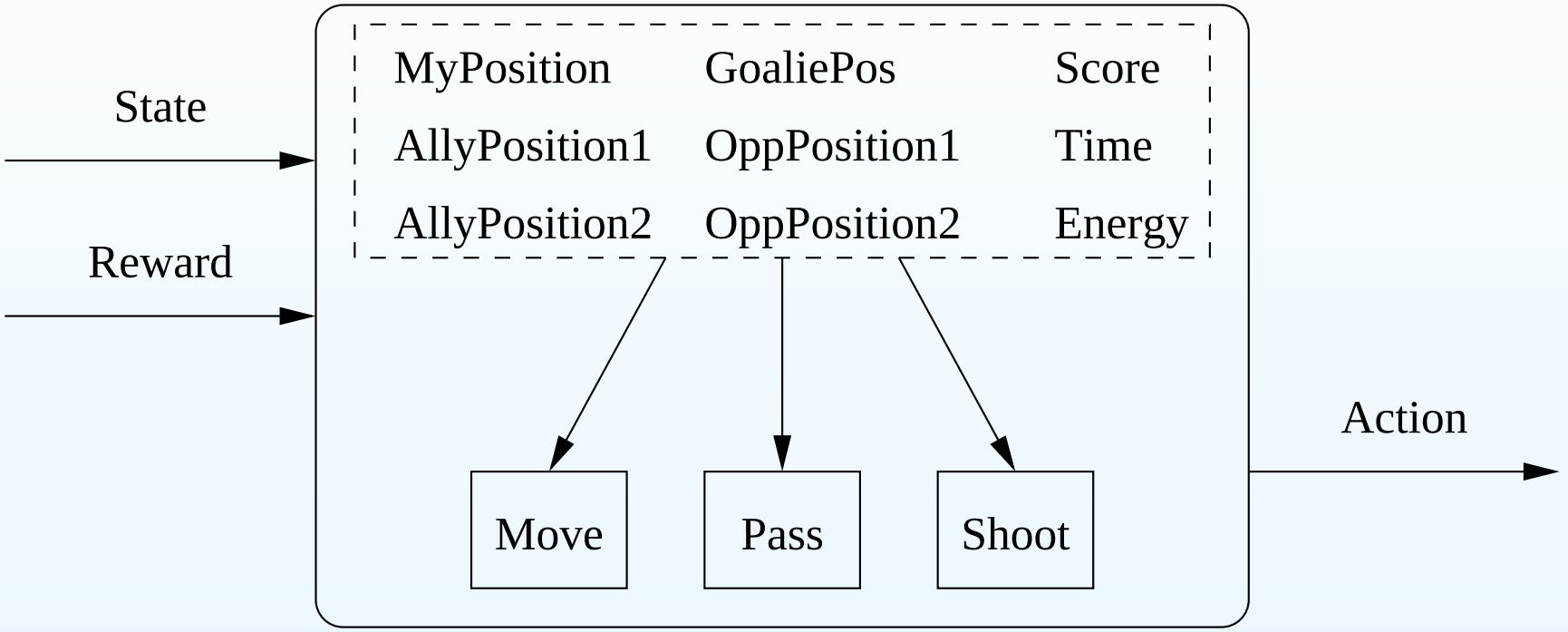
Value-Based RL



Learn: $Q : S \times A \rightarrow \mathbb{R}$

"How much reward can I earn starting at s by choosing a?"

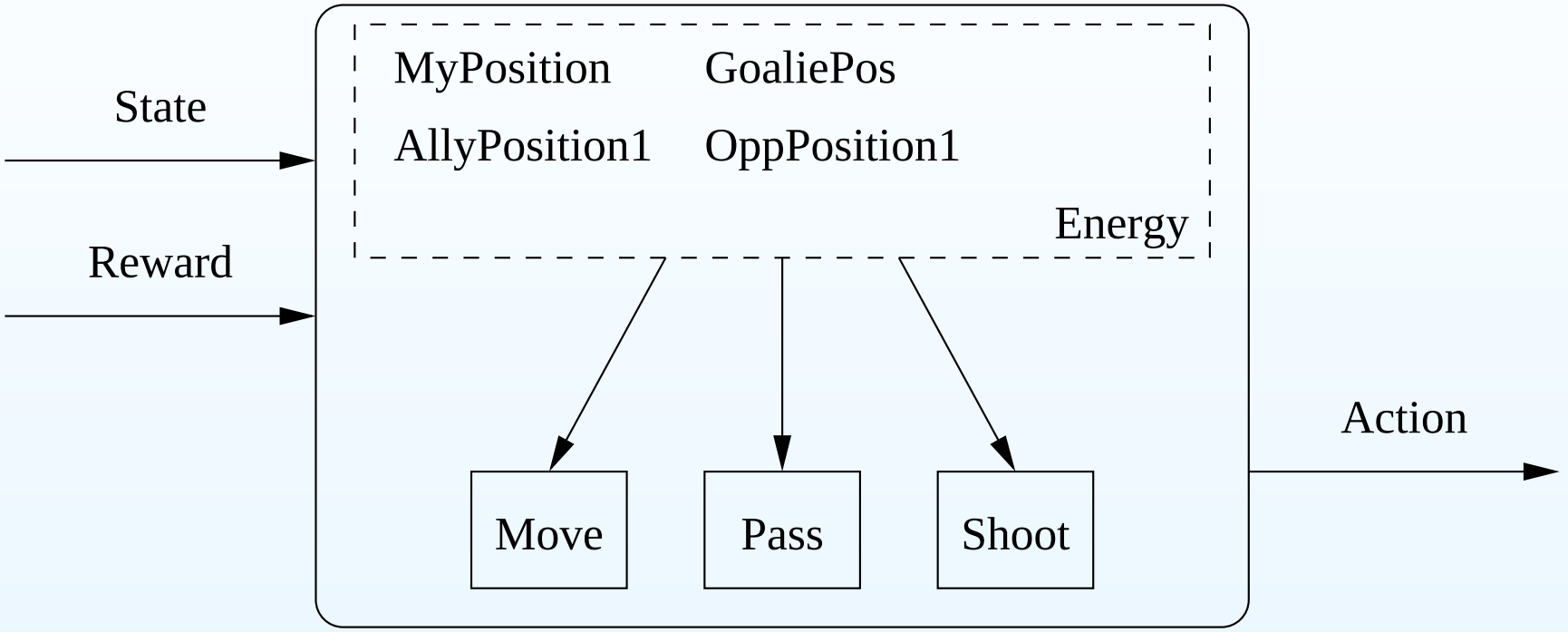
Value-Based RL



Learn: $Q : F_1 \times F_2 \times F_3 \times F_4 \times F_5 \times F_6 \times F_7 \times F_8 \times F_9 \times A \rightarrow \mathbb{R}$

In practice: high-dimensional state spaces

Value-Based RL



Learn: $Q : F_1 \times F_2 \times F_4 \times F_5 \times F_9 \times A \rightarrow \mathbb{R}$

State abstraction: ignore the irrelevant dimensions

State abstraction as qualitative knowledge

- Traditional sources of abstraction
  - Prior knowledge from a human
  - Computation from a given model

State abstraction as qualitative knowledge

- Traditional sources of abstraction
  - Prior knowledge from a human
  - Computation from a given model
- Automatic discovery?
  - But discovering structure is harder than learning policies

State abstraction as qualitative knowledge

- Traditional sources of abstraction
  - Prior knowledge from a human
  - Computation from a given model
- Automatic discovery?
  - But discovering structure is harder than learning policies
  - Our approach: knowledge transfer

1. Discover abstractions in easy domains
2. Transfer abstractions to hard domains

Policy irrelevance: A new basis for state abstraction

When should we ignore a feature?

- Prior work
  - *... if the states share the same abstract one-step model.*
  - Requires the true model of the environment
  - Depends on the global abstraction

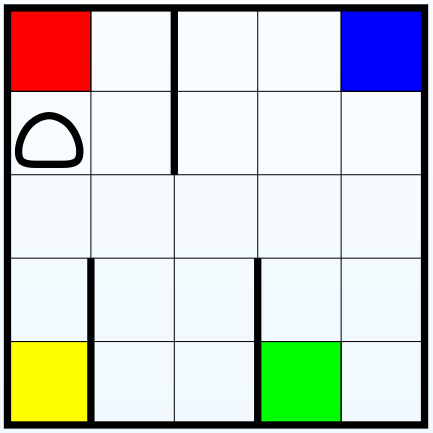
Policy irrelevance: A new basis for state abstraction

When should we ignore a feature?

- Prior work
  - *... if the states share the same abstract one-step model.*
  - Requires the true model of the environment
  - Depends on the global abstraction
- Our work
  - *... if the states share the same optimal action.*
  - Requires a learned policy for the environment
  - Independent of abstraction at other states

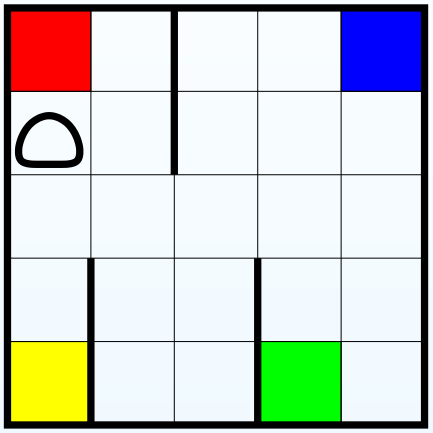
The Taxi domain

- Four features
  - Taxi x coordinate
  - Taxi y coordinate
  - Current passenger location
  - Passenger destination



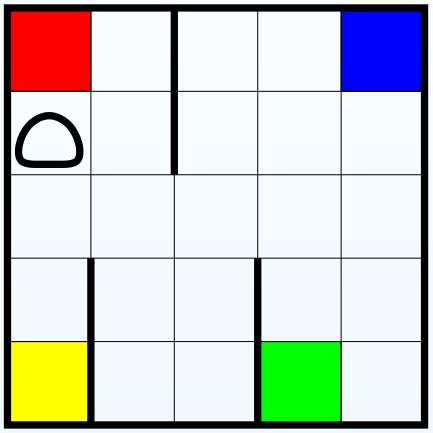
The Taxi domain

- Four features
  - Taxi x coordinate
  - Taxi y coordinate
  - Current passenger location
  - Passenger destination



- Six actions: North, South, East, West, Pick Up, Put Down

The Taxi domain



- Four features
  - Taxi x coordinate
  - Taxi y coordinate
  - Current passenger location
  - Passenger destination
- Six actions: North, South, East, West, Pick Up, Put Down
- Optimal policy:
  - Navigate to the passenger's location
  - Pick up the passenger
  - Navigate to the passenger's destination
  - Put down the passenger

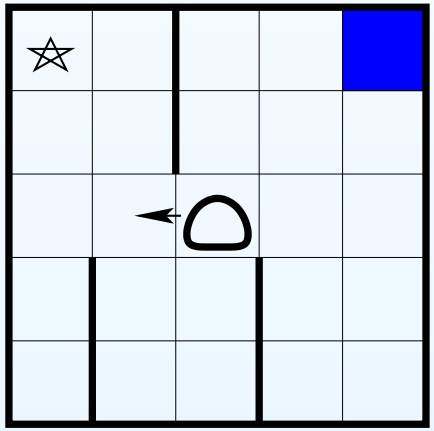
Policy irrelevance in the Taxi domain

Relevance of the passenger destination. . .

Policy irrelevance in the Taxi domain

Relevance of the passenger destination. . .

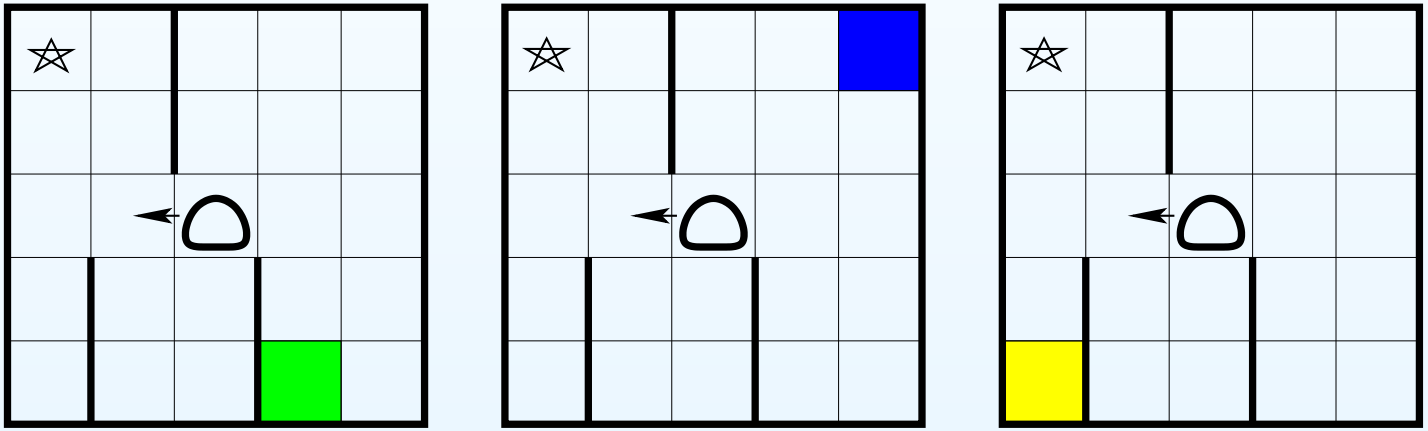
- When the passenger is not inside the taxi



Policy irrelevance in the Taxi domain

Relevance of the passenger destination. . .

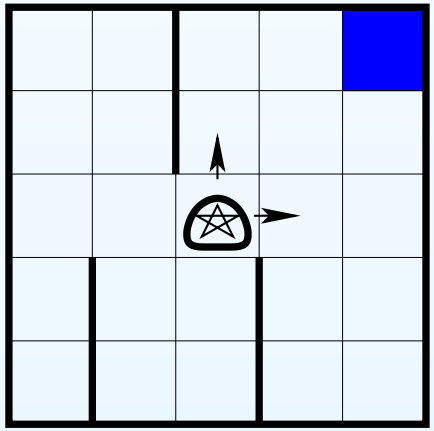
- When the passenger is not inside the taxi



Policy irrelevance in the Taxi domain

Relevance of the passenger destination. . .

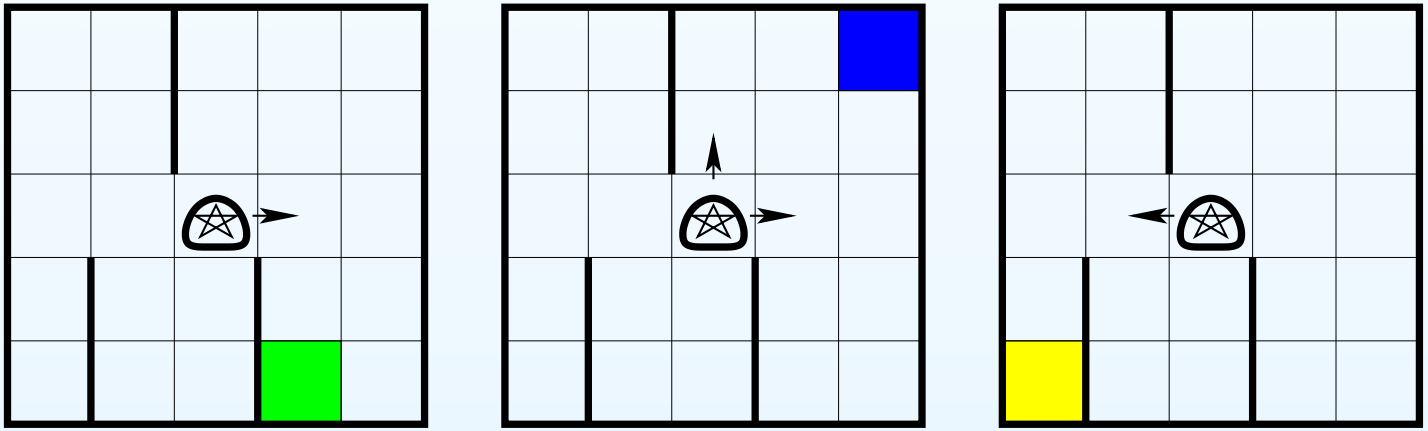
- When the passenger is not inside the taxi
- When the passenger is inside the taxi



Policy irrelevance in the Taxi domain

Relevance of the passenger destination. . .

- When the passenger is not inside the taxi
- When the passenger is inside the taxi



Policy irrelevance with real data

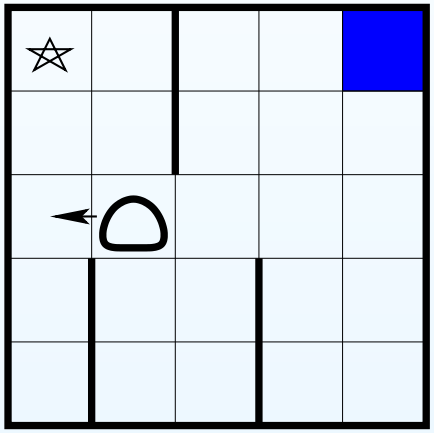
Relevance of the passenger destination. . .

- When the policy is learned from data

Policy irrelevance with real data

Relevance of the passenger destination. . .

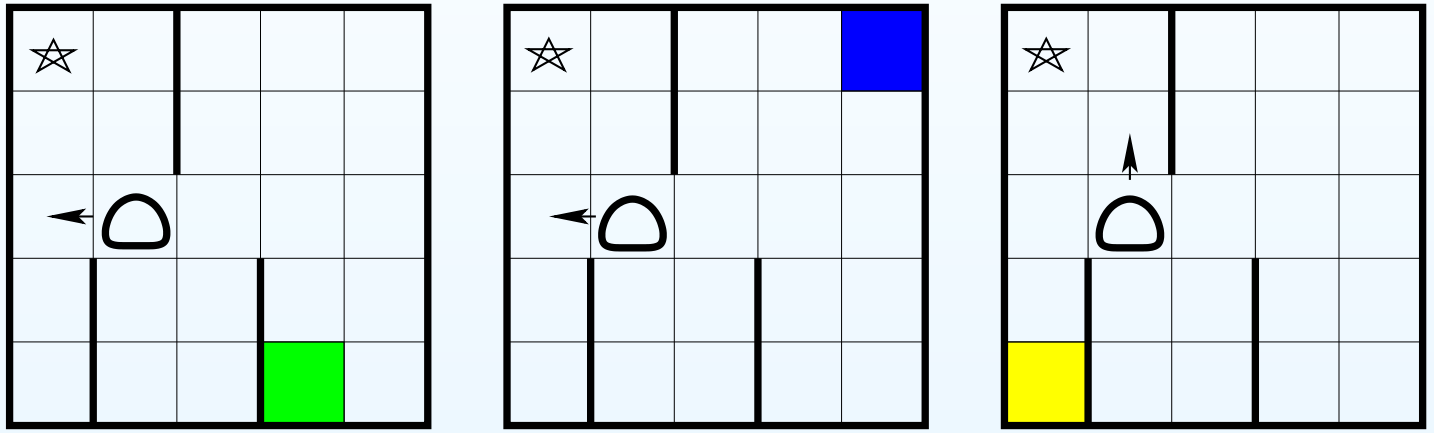
- When the policy is learned from data



Policy irrelevance with real data

Relevance of the passenger destination. . .

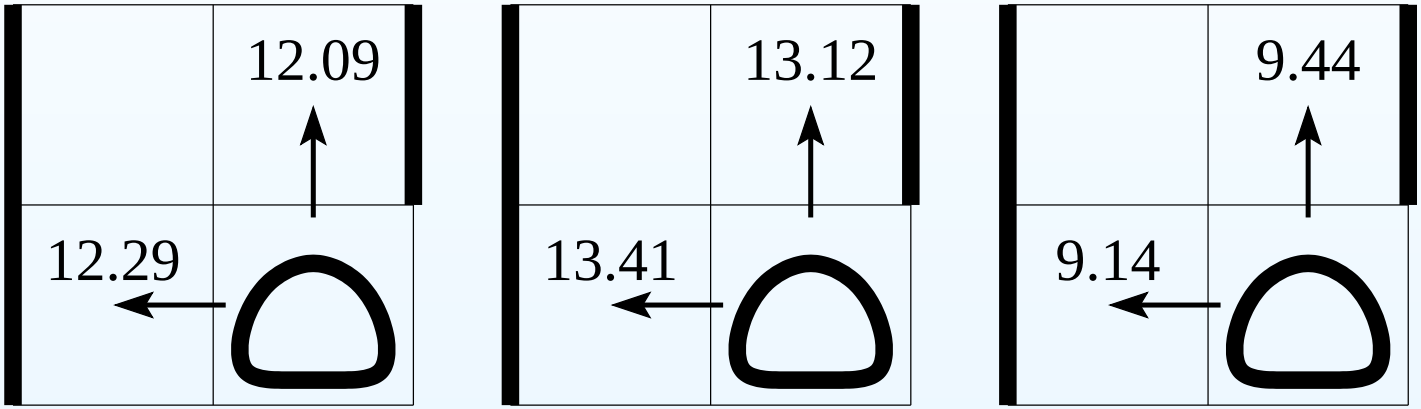
- When the policy is learned from data



Policy irrelevance with real data

Relevance of the passenger destination. . .

- When the policy is learned from data



Policy irrelevance from action-value comparisons

$$Q(s', a) \geq Q(s', a')$$

When should we ignore a set of features F at a state s?

Policy irrelevance from action-value comparisons

$$Q(s', a) \geq Q(s', a')$$

- Action a is better than action a' at state s'

When should we ignore a set of features F at a state s?

Policy irrelevance from action-value comparisons

$$\forall a' \; Q(s', a) \geq Q(s', a')$$

- Action a is better than action a' at state s'
- Action a is optimal at state s'

When should we ignore a set of features F at a state s?

Policy irrelevance from action-value comparisons

$$\forall s' \in [s]_F \; \forall a' \; Q(s', a) \geq Q(s', a')$$

- Action a is better than action a' at state s'
- Action a is optimal at state s'
- Action a is optimal at every state $s' \in [s]_F$

When should we ignore a set of features F at a state s?

($[s]_F$ is the set of states obtained from s by varying over F)

Policy irrelevance from action-value comparisons

$$\exists a \, \forall s' \in [s]_F \, \forall a' \, Q(s', a) \geq Q(s', a')$$

- Action a is better than action a' at state s'
- Action a is optimal at state s'
- Action a is optimal at every state $s' \in [s]_F$
- Some action is optimal at every $s' \in [s]_F$

When should we ignore a set of features F at a state s?

($[s]_F$ is the set of states obtained from s by varying over F)

Policy irrelevance from action-value comparisons

$$\exists a \; \forall s' \in [s]_F \; \forall a' \; Q(s', a) \geq Q(s', a')$$

- Action a is better than action a' at state s'
- Action a is optimal at state s'
- Action a is optimal at every state $s' \in [s]_F$
- Some action is optimal at every $s' \in [s]_F$
- Features F are policy irrelevant at s

When should we ignore a set of features F at a state s?

($[s]_F$ is the set of states obtained from s by varying over F)

Robust action-value comparison via sampling

$$Q(s', a) \stackrel{?}{\geq} Q(s', a')$$

Robust action-value comparison via sampling

$$Q(s', a) \stackrel{?}{\geq} Q(s', a')$$

- Compare samples of estimates, not individual estimates!

Robust action-value comparison via sampling

$$Q(s', a) \overset{?}{\geq} Q(s', a')$$

- Compare samples of estimates, not individual estimates!
- Method 1: Statistical hypothesis testing
  - Solve task repeatedly with a value-based RL algorithm
  - Low computational but high sample complexity

Robust action-value comparison via sampling

$$Q(s', a) \stackrel{?}{\geq} Q(s', a')$$

- Compare samples of estimates, not individual estimates!
- Method 1: Statistical hypothesis testing
  - Solve task repeatedly with a value-based RL algorithm
  - Low computational but high sample complexity
- Method 2: Monte Carlo simulation
  - Construct a Bayesian model from an experience trace
  - Low sample but high computational complexity

Partial state abstractions

Are features F relevant at state s?

At what states is each set of features relevant?

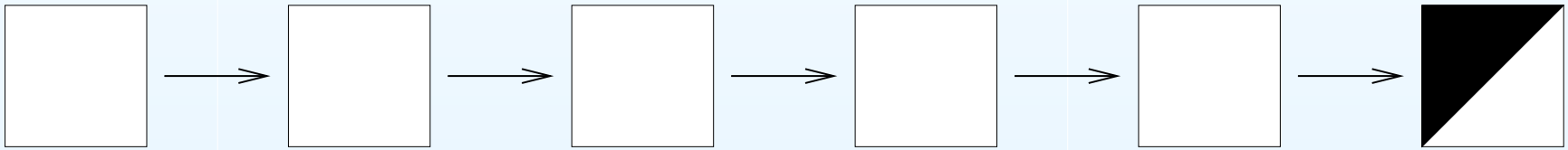
- Train a binary classifier for certain sets of features
- Learn when each set of features is irrelevant
- Naive application: ignore F at classified states

Transferring abstractions to novel domains

- Sources of error for straightforward state aggregation
  - Statistical testing error
  - Generalization error of the learned classifiers
  - Novelty in the transfer domain
  - Disruption of value-function semantics!

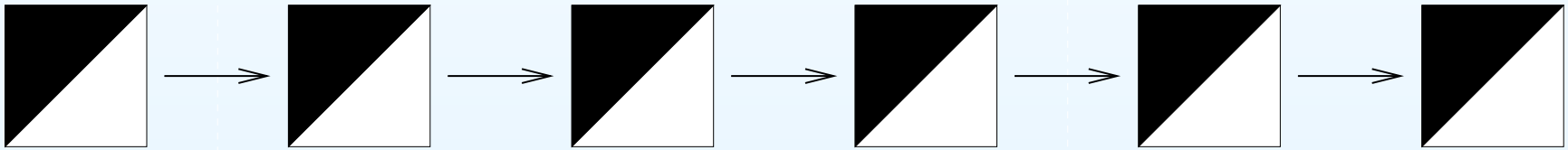
Transferring abstractions to novel domains

- Sources of error for straightforward state aggregation
  - Statistical testing error
  - Generalization error of the learned classifiers
  - Novelty in the transfer domain
  - Disruption of value-function semantics!



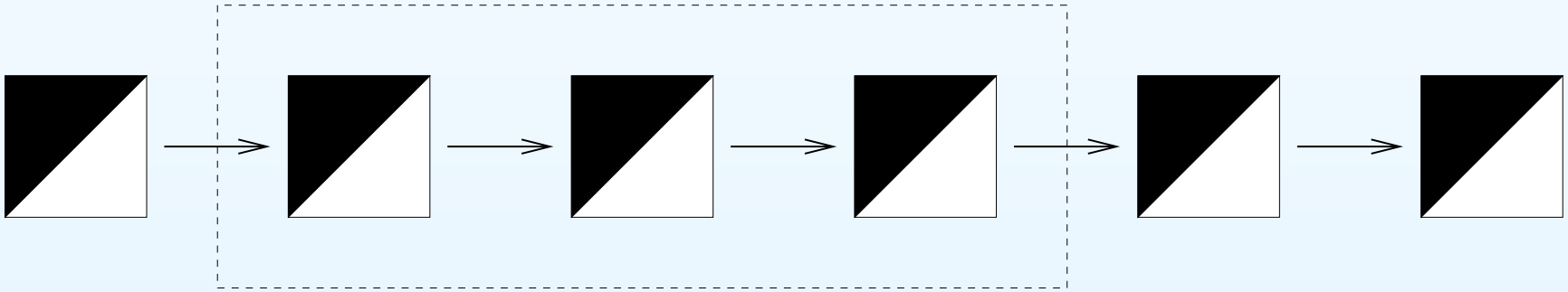
Transferring abstractions to novel domains

- Sources of error for straightforward state aggregation
  - Statistical testing error
  - Generalization error of the learned classifiers
  - Novelty in the transfer domain
  - Disruption of value-function semantics!



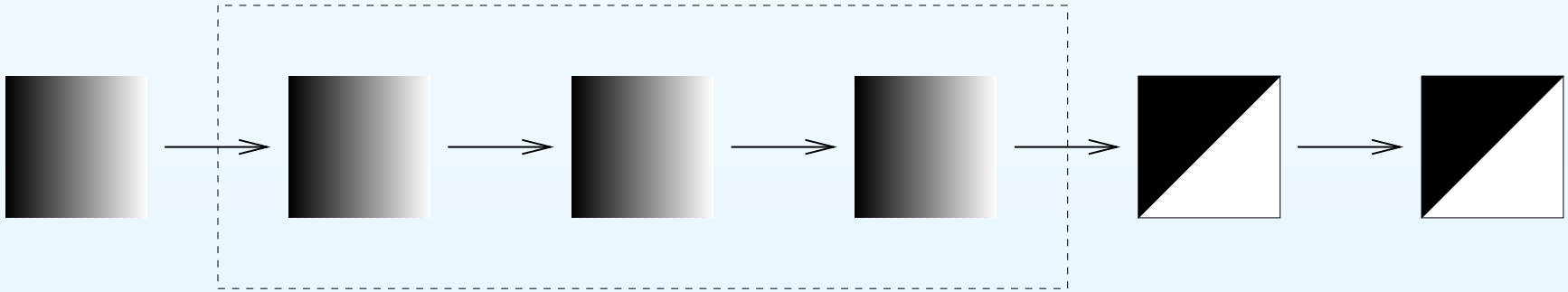
Transferring abstractions to novel domains

- Sources of error for straightforward state aggregation
  - Statistical testing error
  - Generalization error of the learned classifiers
  - Novelty in the transfer domain
  - Disruption of value-function semantics!



Transferring abstractions to novel domains

- Sources of error for straightforward state aggregation
  - Statistical testing error
  - Generalization error of the learned classifiers
  - Novelty in the transfer domain
  - Disruption of value-function semantics!

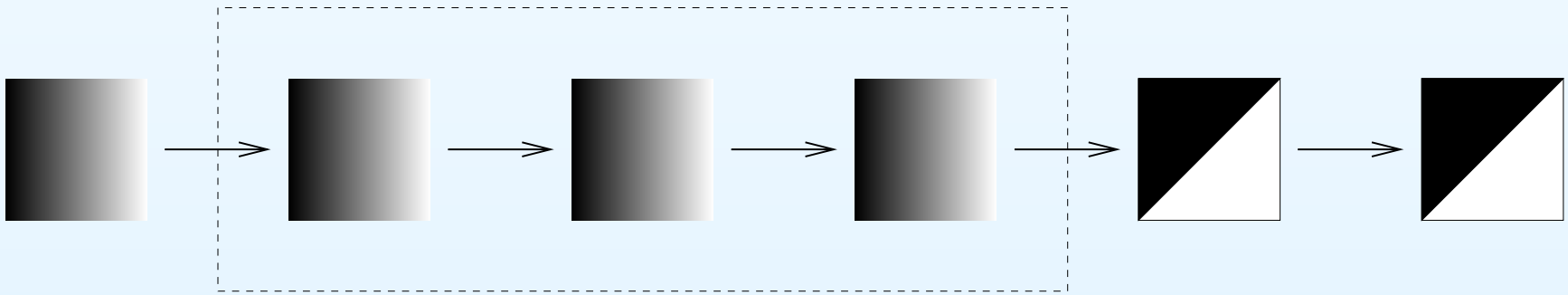


Temporal abstraction

- One abstract action comprises a sequence of actions
- AKA subroutines, options, subtasks

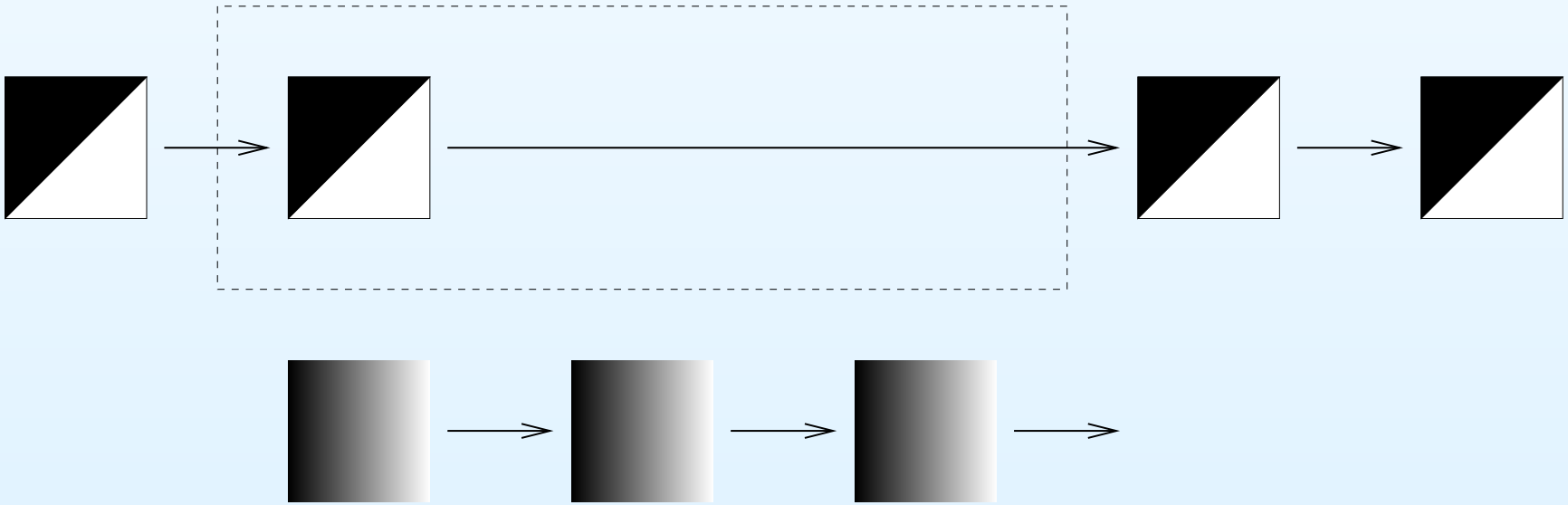
Temporal abstraction

- One abstract action comprises a sequence of actions
- AKA subroutines, options, subtasks



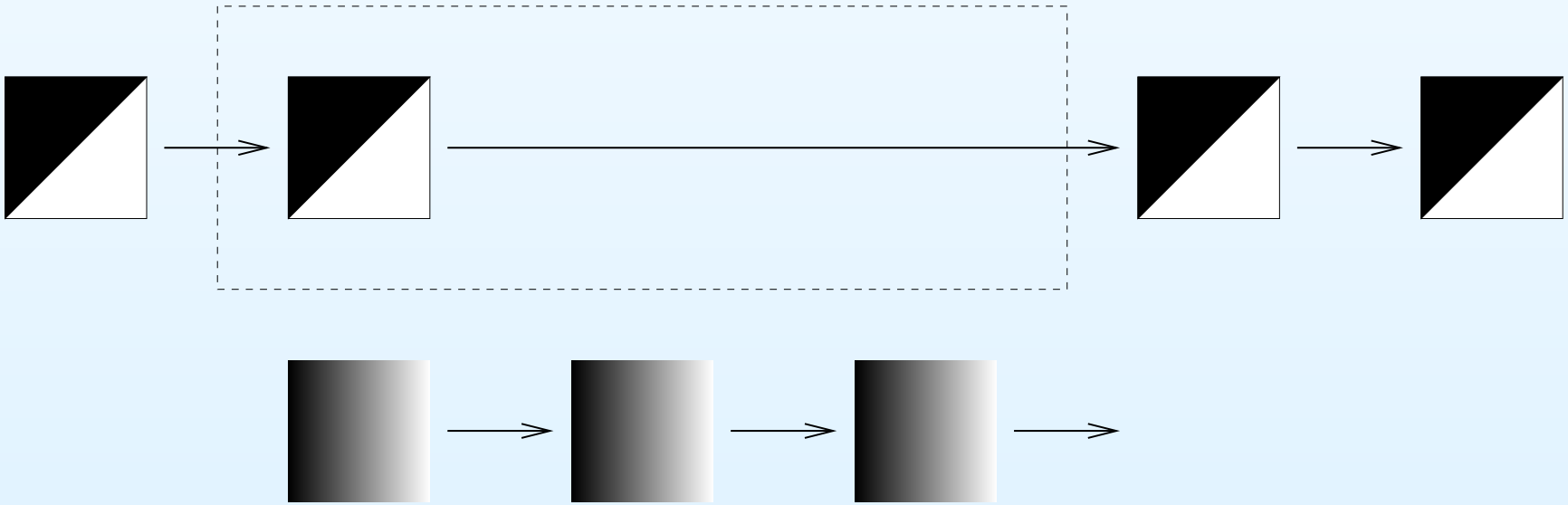
Temporal abstraction

- One abstract action comprises a sequence of actions
- AKA subroutines, options, subtasks



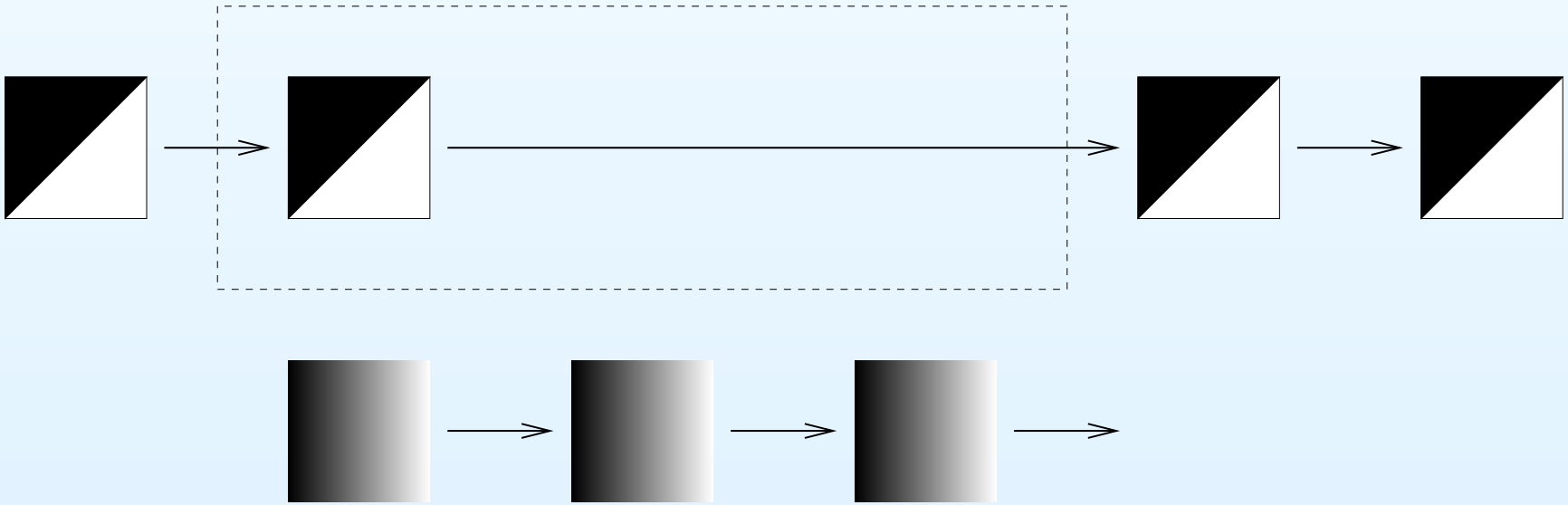
Temporal abstraction

- One abstract action comprises a sequence of actions
- AKA subroutines, options, subtasks
- Prior research: “Achieve this subgoal state”
- Our research: “Ignore these features”

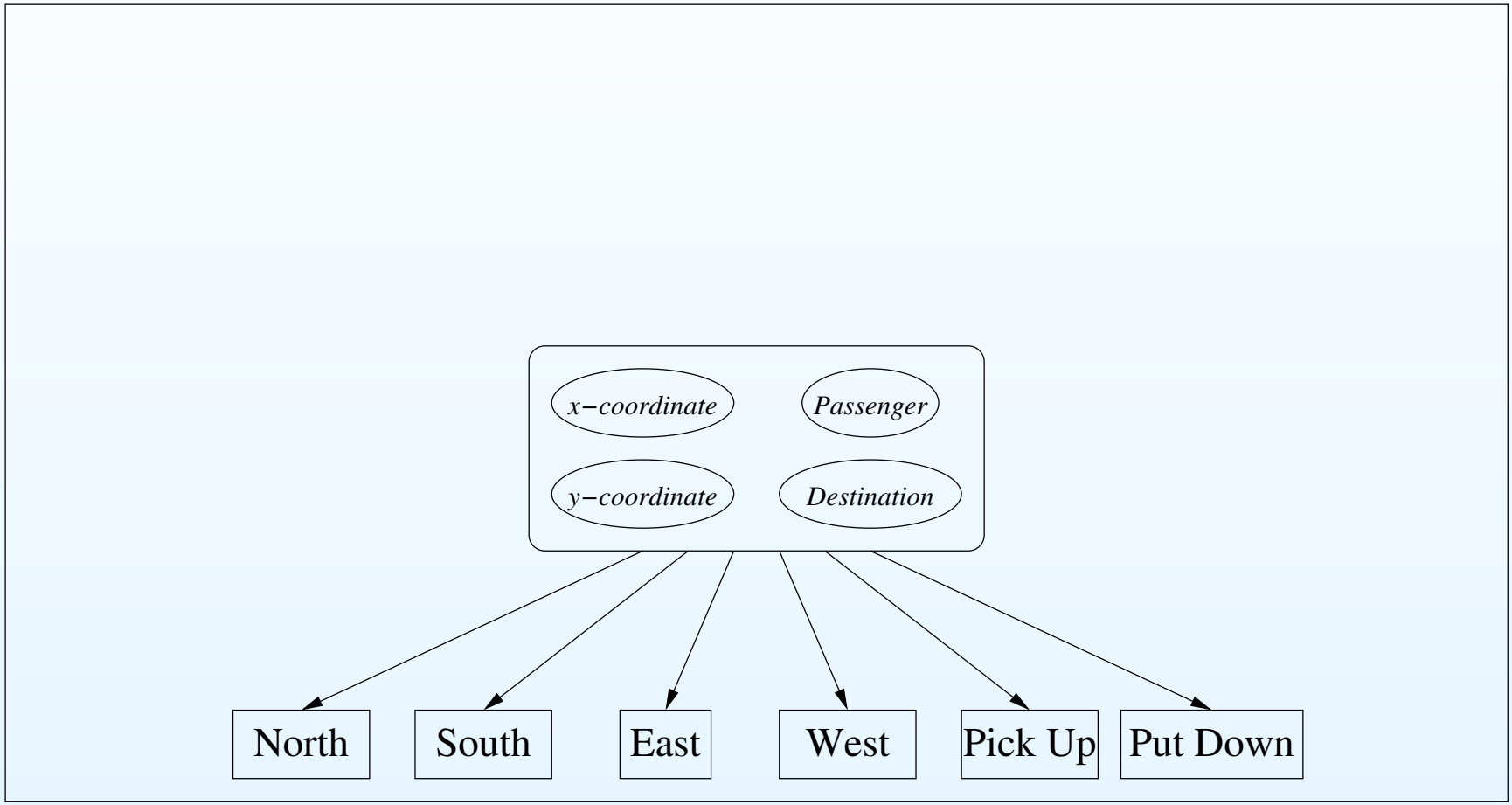


Temporal abstraction

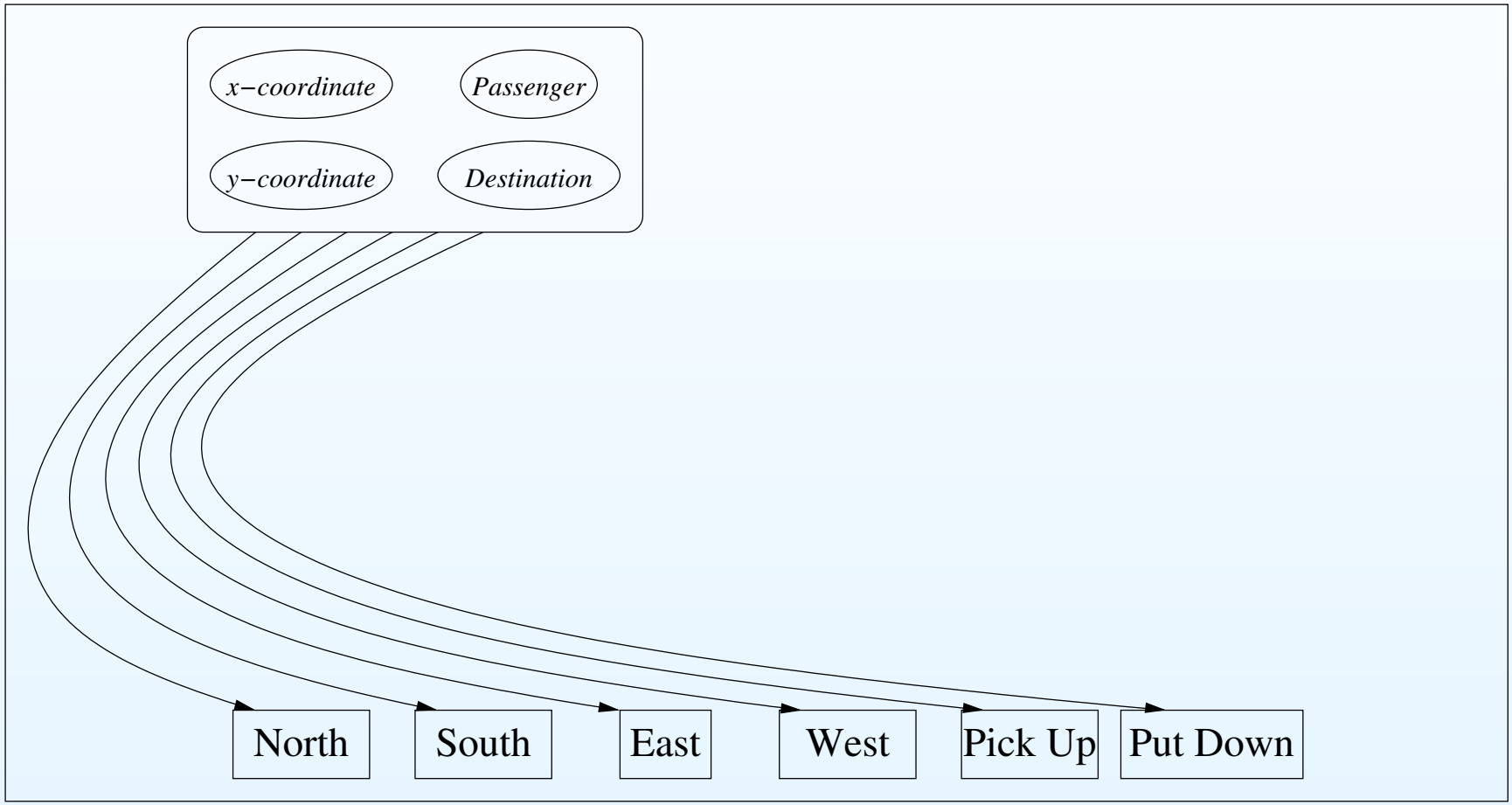
- One abstract action comprises a sequence of actions
- AKA subroutines, options, subtasks
- Prior research: “Achieve this subgoal state”
- Our research: “Ignore these features”
- Safe encapsulation of state abstractions into actions
- Learn when to apply discovered state abstractions!



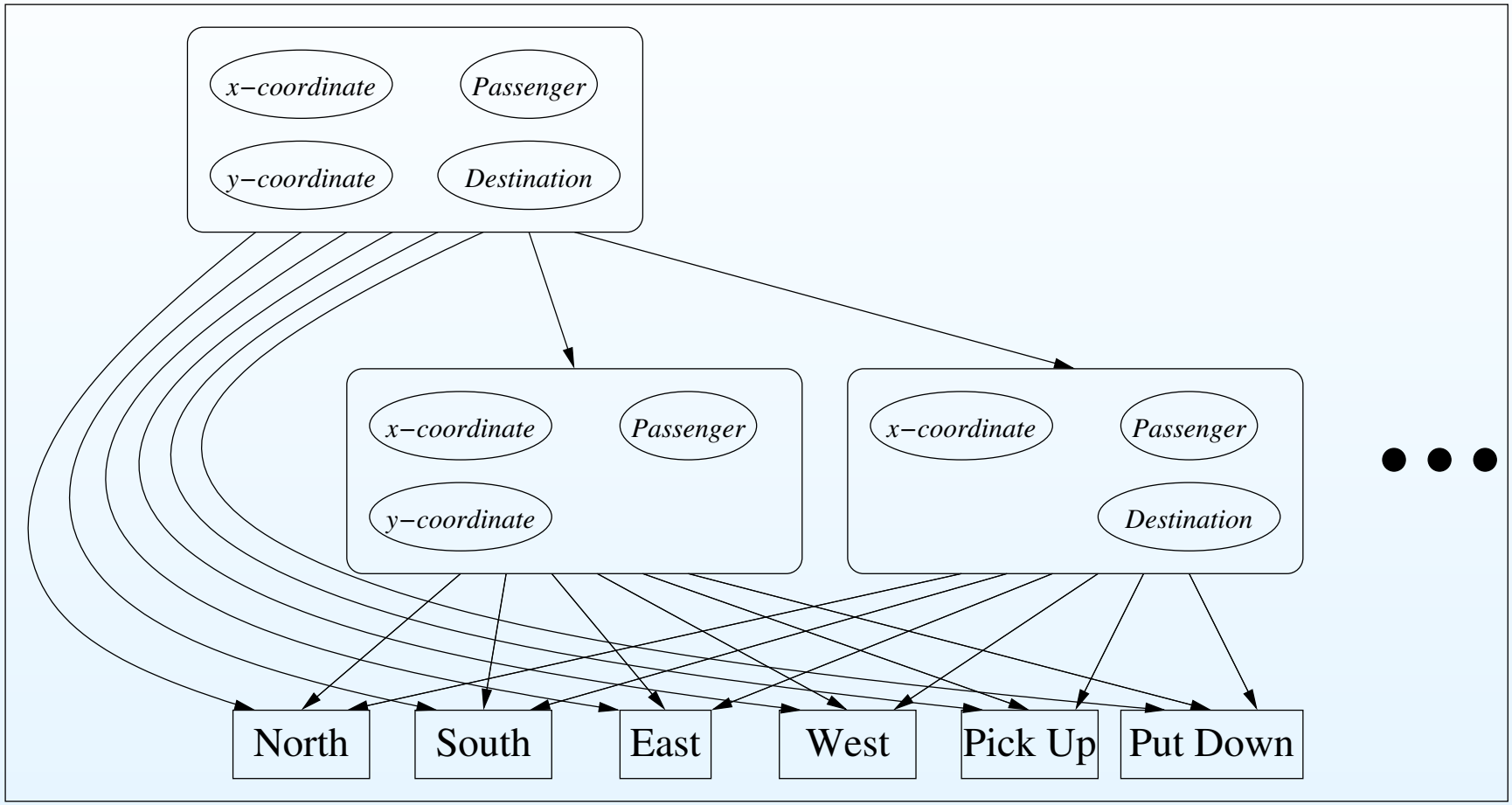
Hierarchies of state and temporal abstractions



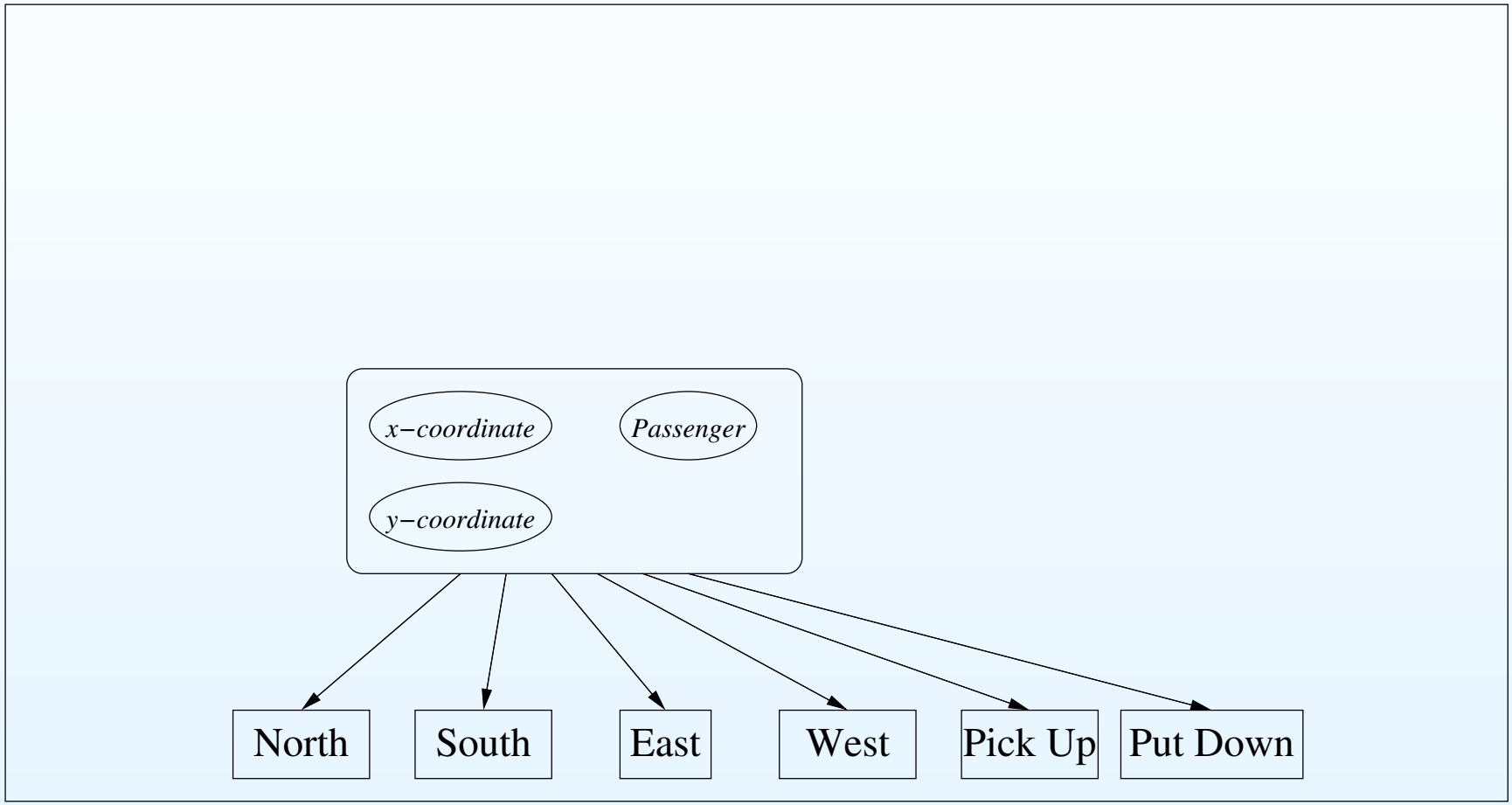
Hierarchies of state and temporal abstractions



Hierarchies of state and temporal abstractions

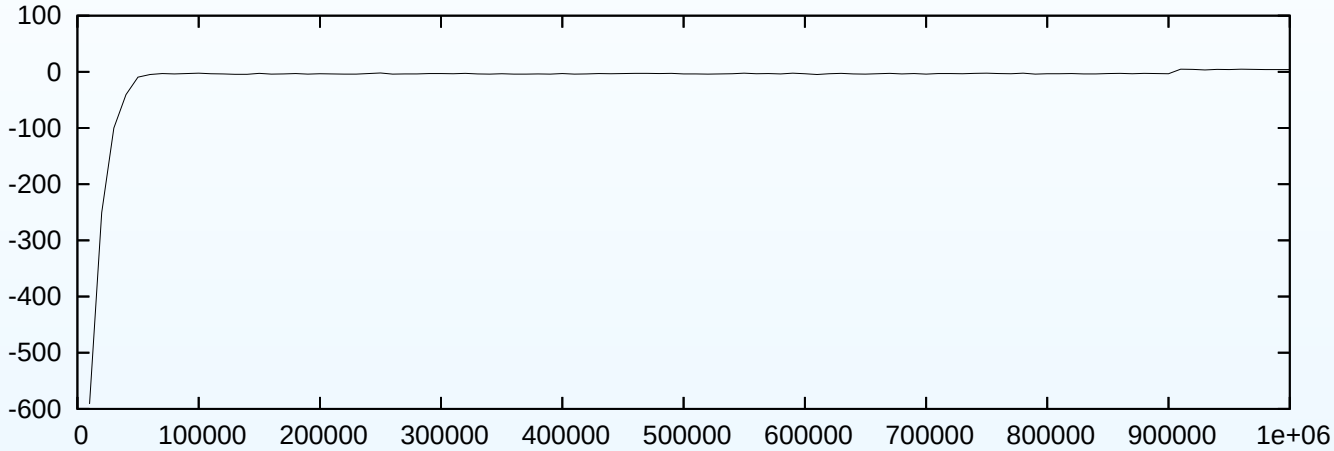


Hierarchies of state and temporal abstractions



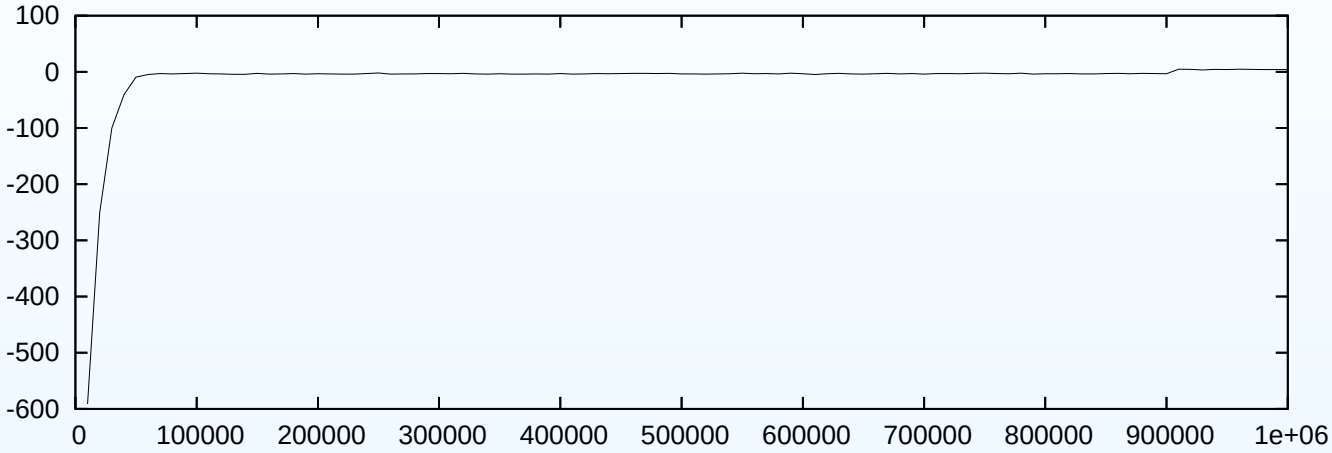
Results in the Taxi domain

- Original 5×5 domain

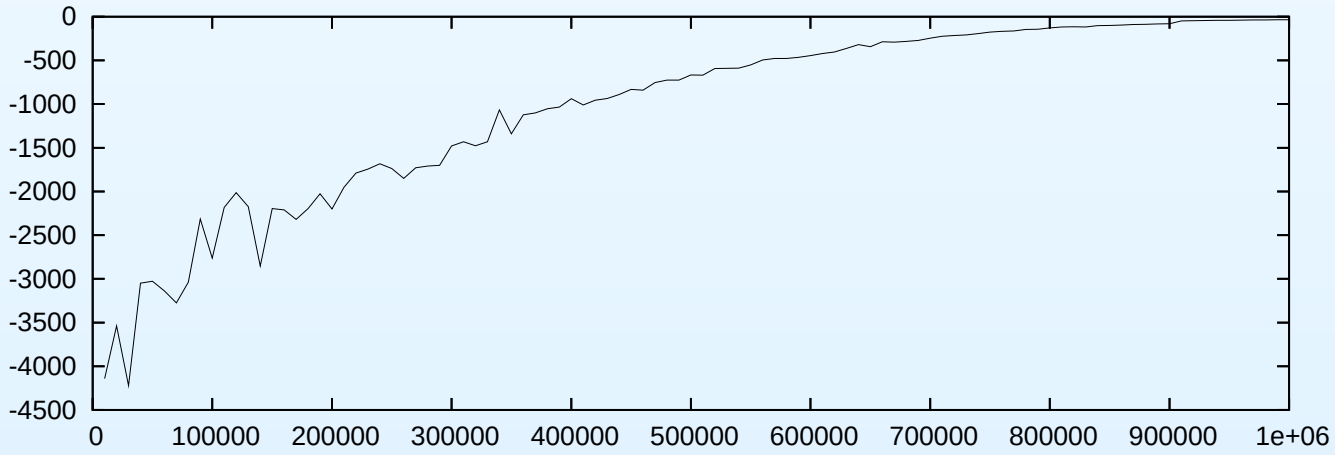


Results in the Taxi domain

- Original 5×5 domain

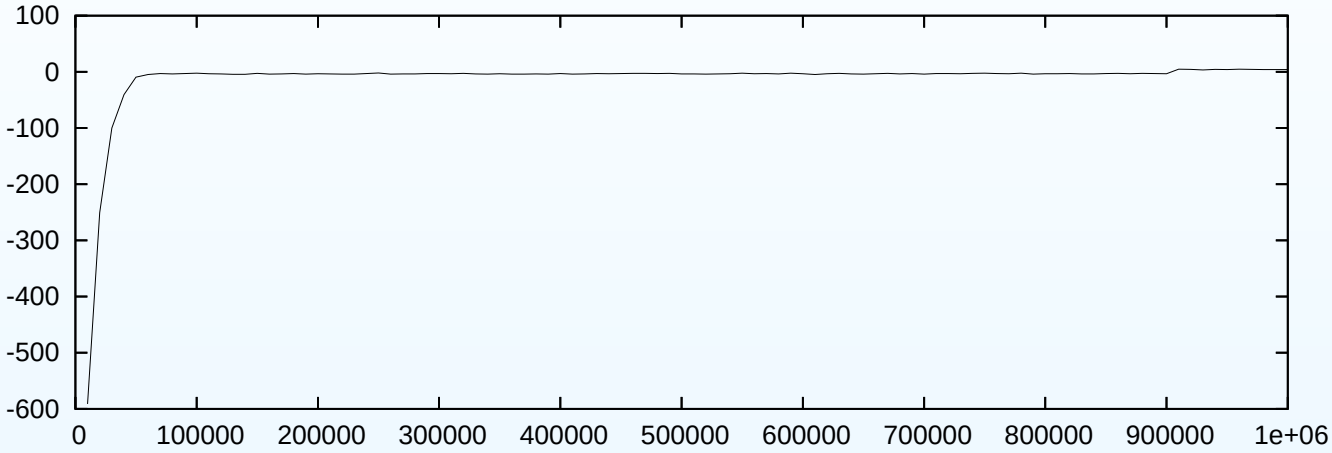


- Randomly generated 10×10 domain

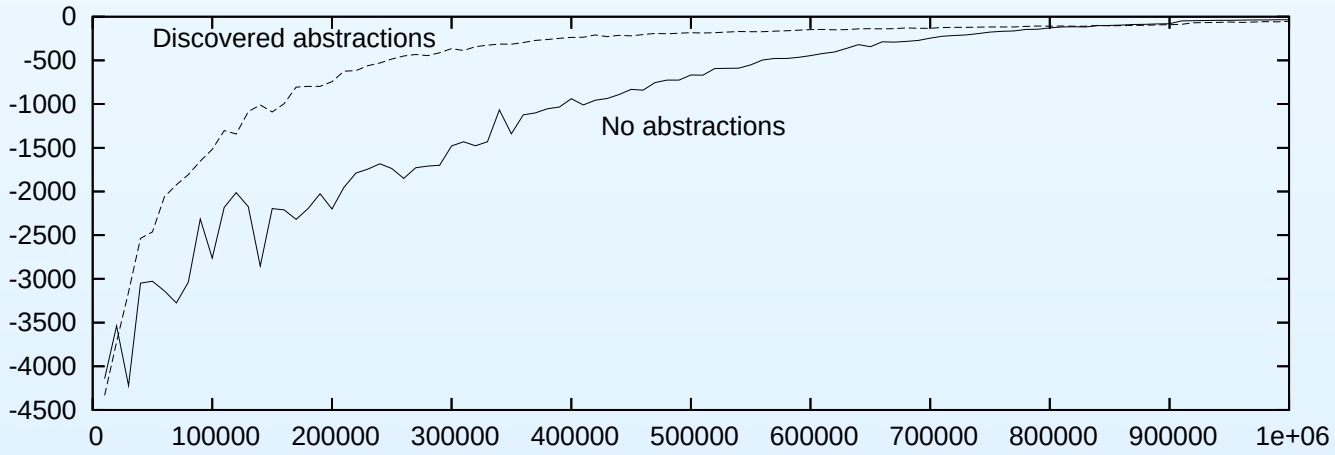


Results in the Taxi domain

- Original 5×5 domain



- Randomly generated 10×10 domain



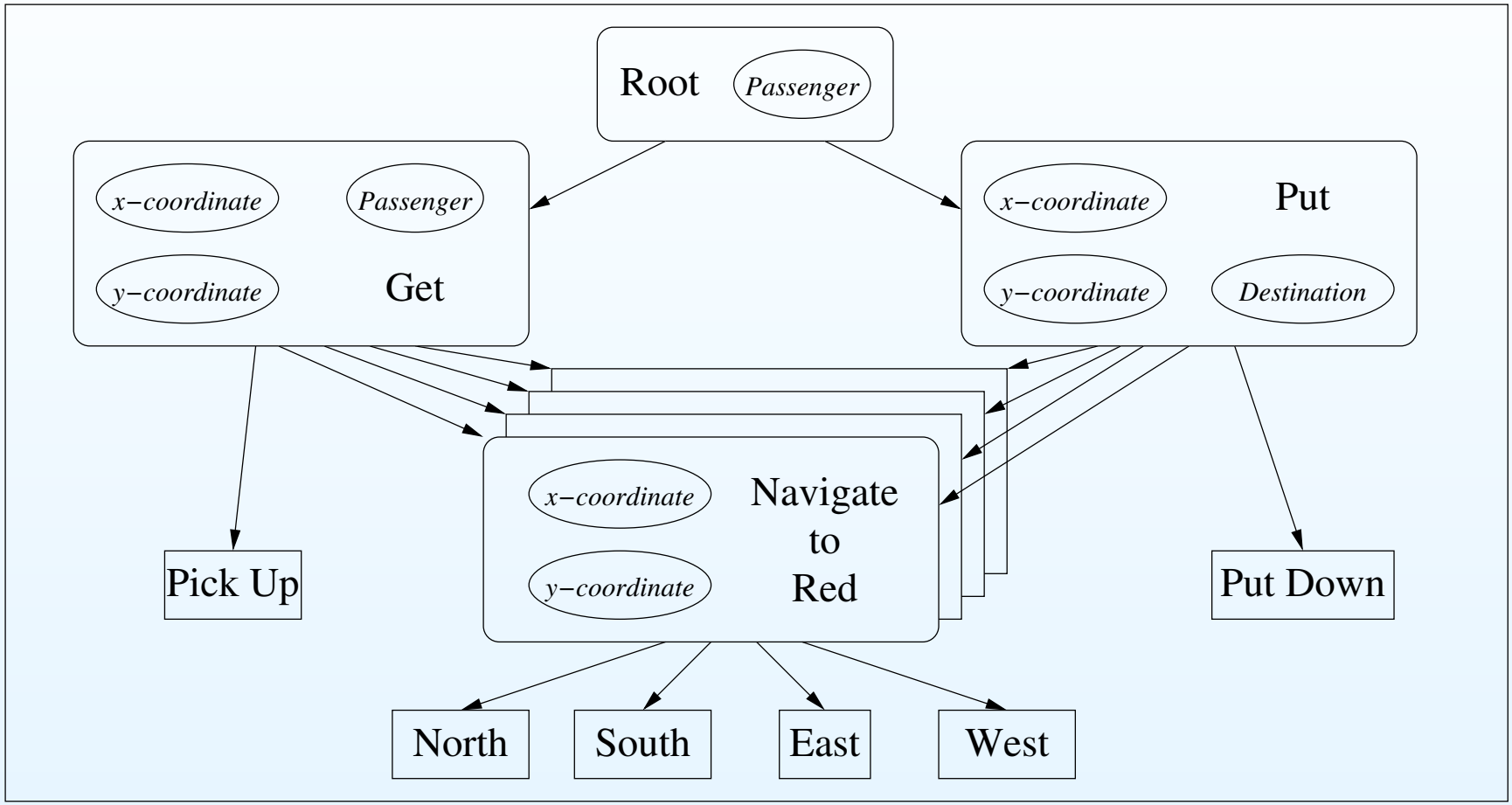
Conclusions

- Abstraction discovery as problem reformulation
- A new basis for state abstraction: policy irrelevance
  - Statistical testing methods
  - Trajectory-based discovery algorithm
- Safe transfer of state abstractions to novel domains
  - Encapsulation inside temporal abstractions
  - Synergy of temporal and state abstractions

Future work

- Adjusting abstraction-termination conditions
- Detection of dynamic domains
- Application to larger domains
  - Function approximation
  - Model-based RL algorithms
- Recursive abstraction discovery
  - Discovery of hierarchy
  - Dynamic state abstraction

Future work: discovery of hierarchy



The discovery algorithm

Which feature sets F to test at what states s?

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune

s

F_1

F_2

F_3

$F_{1,2}$

$F_{1,3}$

$F_{2,3}$

$F_{1,2,3}$

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune

	s
F_1	yes
F_2	no
F_3	yes
$F_{1,2}$	
$F_{1,3}$	
$F_{2,3}$	
$F_{1,2,3}$	

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune

	s
F_1	yes
F_2	no
F_3	yes
$F_{1,2}$	no
$F_{1,3}$	
$F_{2,3}$	no
$F_{1,2,3}$	no

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune

	s
F_1	yes
F_2	no
F_3	yes
$F_{1,2}$	no
$F_{1,3}$	no
$F_{2,3}$	no
$F_{1,2,3}$	no

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune
- Sample states s from solution trajectories

	s_1	s_2	s_3	s_4	s_5	s_6	$\cdots$
F_1	yes						
F_2	no						
F_3	yes						
$F_{1,2}$	no						
$F_{1,3}$	no						
$F_{2,3}$	no						
$F_{1,2,3}$	no						

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune
- Sample states s from solution trajectories

	s_1	s_2	s_3	s_4	s_5	s_6	$\cdots$
F_1	yes	yes	yes	yes	no	no	$\cdots$
F_2	no	no	no	yes	no	yes	$\cdots$
F_3	yes	no	no	yes	yes	yes	$\cdots$
$F_{1,2}$	no	no	no	no	no	no	$\cdots$
$F_{1,3}$	no	no	no	yes	no	no	$\cdots$
$F_{2,3}$	no	no	no	no	no	no	$\cdots$
$F_{1,2,3}$	no	no	no	no	no	no	$\cdots$

The discovery algorithm

Which feature sets F to test at what states s?

- For given state s, test small feature sets F first and prune
- Sample states s from solution trajectories
- Construct a binary classification problem for each F

	s_1	s_2	s_3	s_4	s_5	s_6	$\cdots$
F_1	yes	yes	yes	yes	no	no	$\cdots$
F_2	no	no	no	yes	no	yes	$\cdots$
F_3	yes	no	no	yes	yes	yes	$\cdots$
$F_{1,2}$	no	no	no	no	no	no	$\cdots$
$F_{1,3}$	no	no	no	yes	no	no	$\cdots$
$F_{2,3}$	no	no	no	no	no	no	$\cdots$
$F_{1,2,3}$	no	no	no	no	no	no	$\cdots$

Some abstractions discovered in the Taxi domain

1. Taxi's x-coordinate:
   (a) $y = 1 \wedge$ passenger in taxi $\wedge$ destination Red $\Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*
2. Taxi's y-coordinate:
   (a) $x = 4 \wedge$ passenger in taxi $\Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*
3. Passenger's destination:
   (a) passenger in taxi $\Rightarrow$ *relevant*
   (b) otherwise, *irrelevant*
4. Passenger's location *and* destination:
   (a) $(x = 1 \wedge y = 2) \vee (x = 1 \wedge y = 1) \Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*

Some abstractions discovered in the Taxi domain

1. Taxi's x-coordinate:
   (a) $y = 1 \wedge$ passenger in taxi $\wedge$ destination Red $\Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*
2. Taxi's y-coordinate:
   (a) $x = 4 \wedge$ passenger in taxi $\Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*
3. Passenger's destination: GOOD
   (a) passenger in taxi $\Rightarrow$ *relevant*
   (b) otherwise, *irrelevant*
4. Passenger's location *and* destination:
   (a) $(x = 1 \wedge y = 2) \vee (x = 1 \wedge y = 1) \Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*

Some abstractions discovered in the Taxi domain

1. Taxi's x-coordinate: BAD: testing or classification error!
   (a) $y = 1 \wedge$ passenger in taxi $\wedge$ destination Red $\Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*
2. Taxi's y-coordinate: BAD: testing or classification error!
   (a) $x = 4 \wedge$ passenger in taxi $\Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*
3. Passenger's destination: GOOD
   (a) passenger in taxi $\Rightarrow$ *relevant*
   (b) otherwise, *irrelevant*
4. Passenger's location *and* destination: BAD: task-specific!
   (a) $(x = 1 \wedge y = 2) \vee (x = 1 \wedge y = 1) \Rightarrow$ *irrelevant*
   (b) otherwise, *relevant*