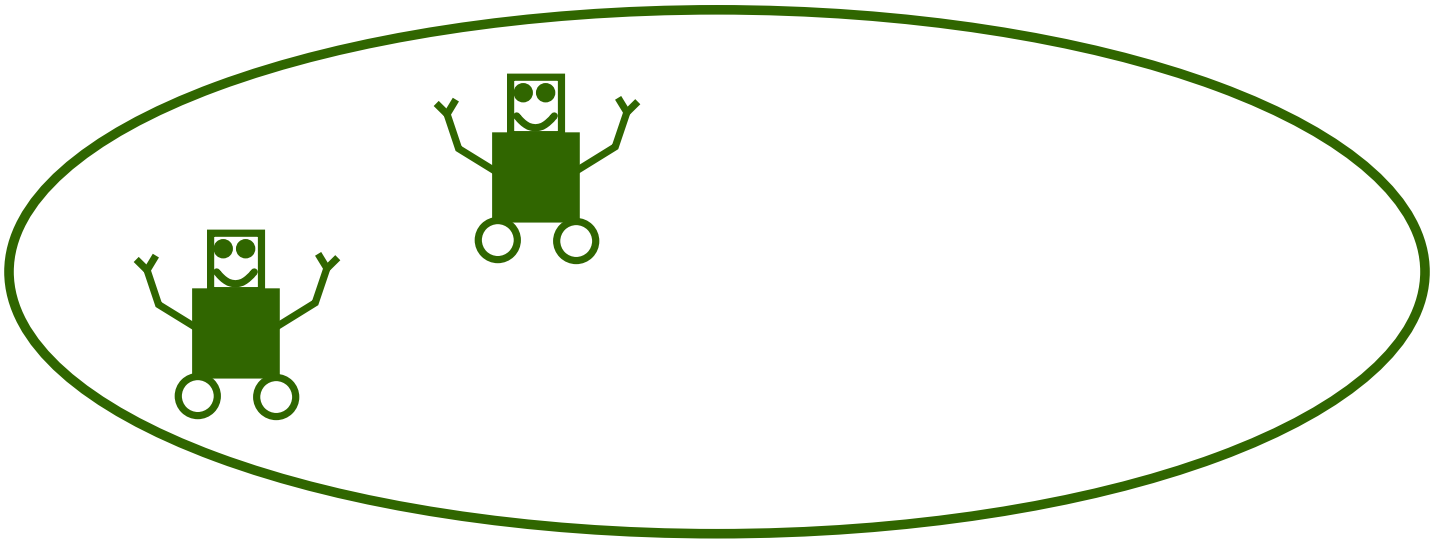
Coaching: Learning and Using Environment and Agent Models for Advice

Patrick Riley

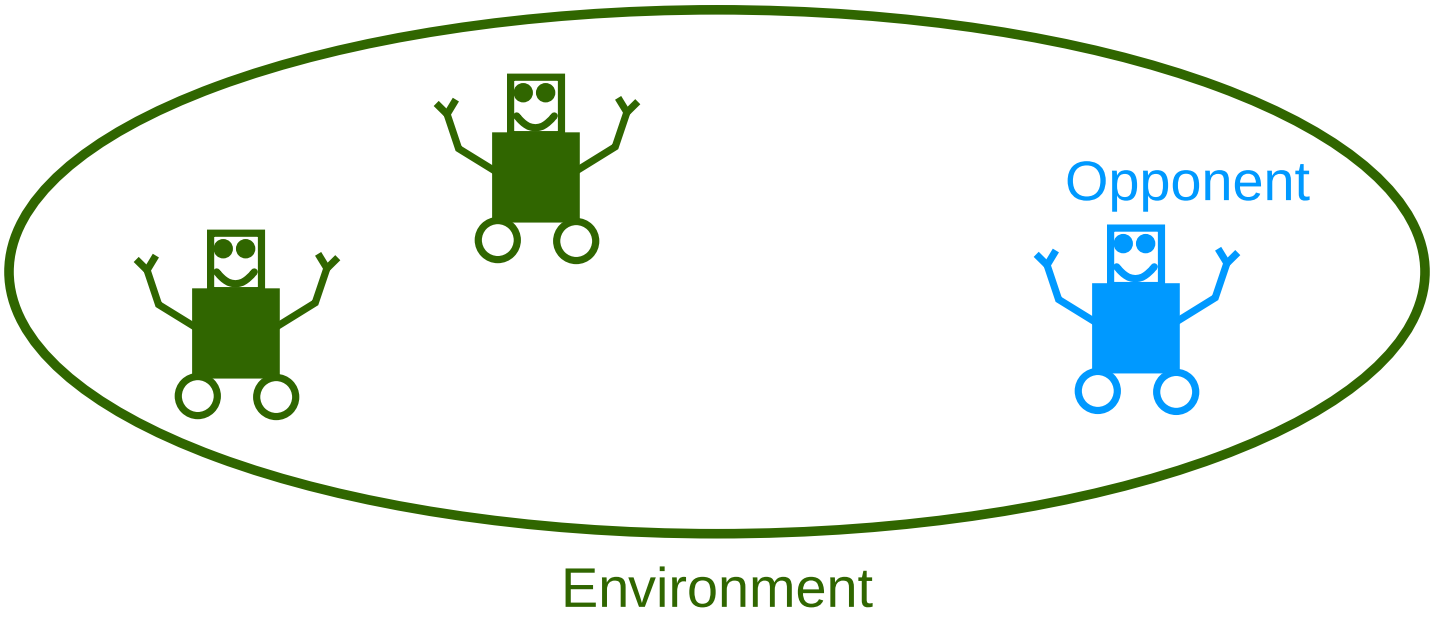
February 1, 2005

Thesis Committee:
Manuela Veloso, Chair
Tom Mitchell
Jack Mostow
Milind Tambe, University of Southern California

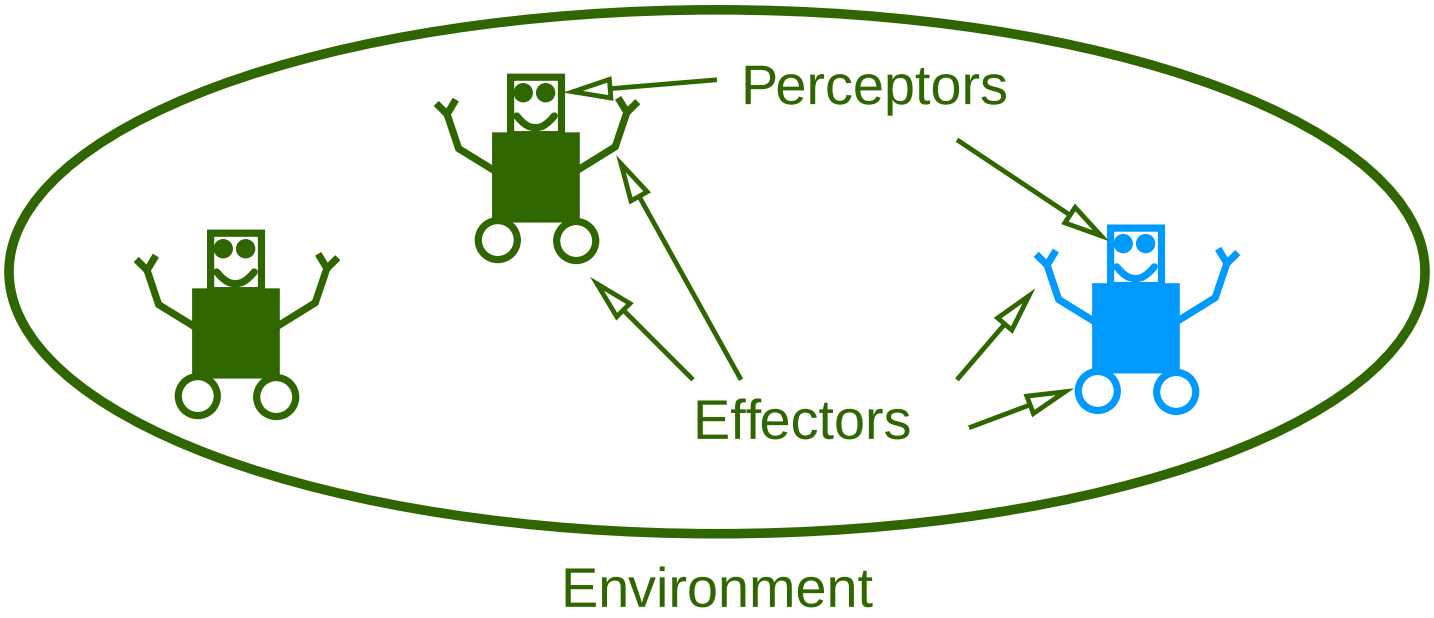
Coaching?



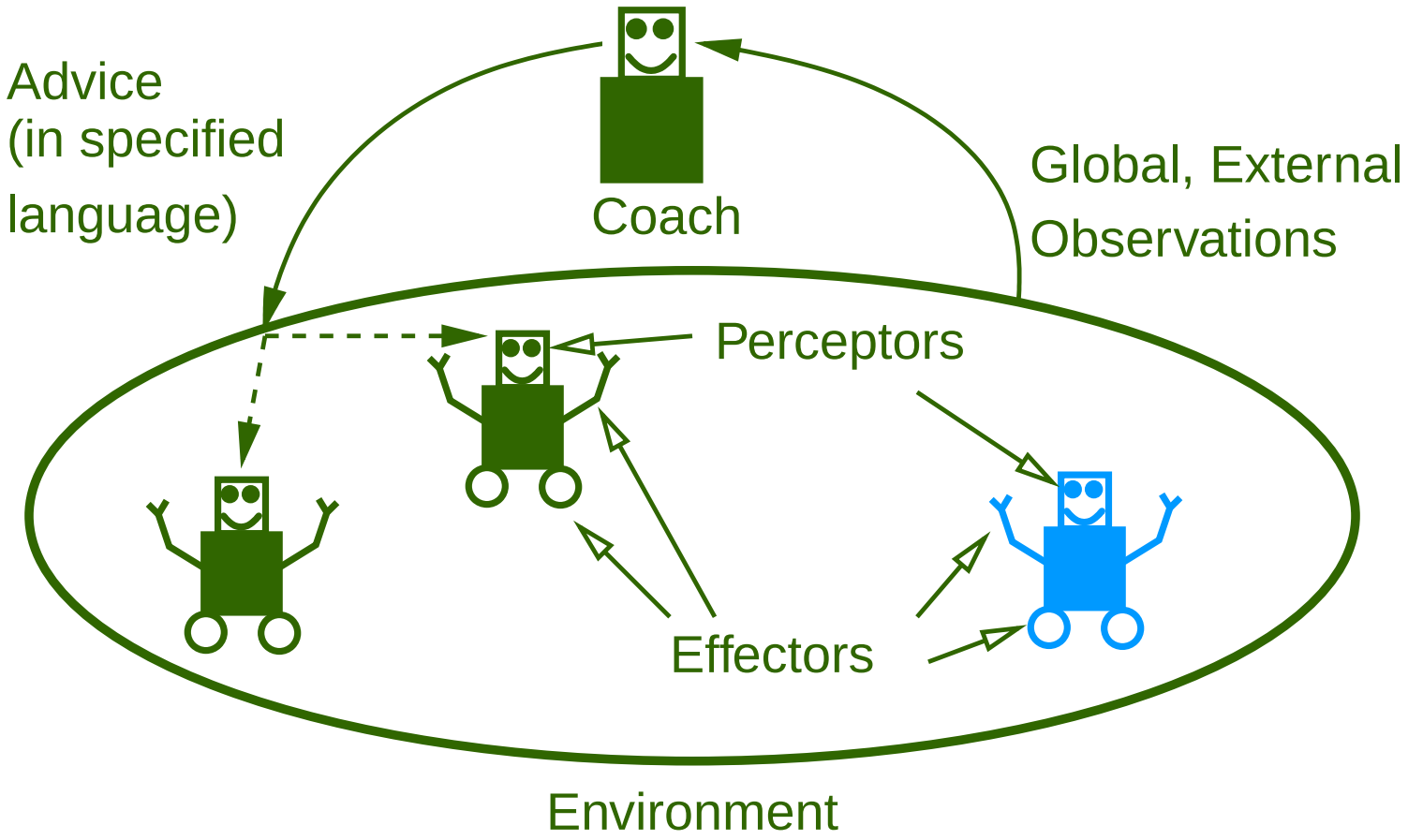
Coaching?



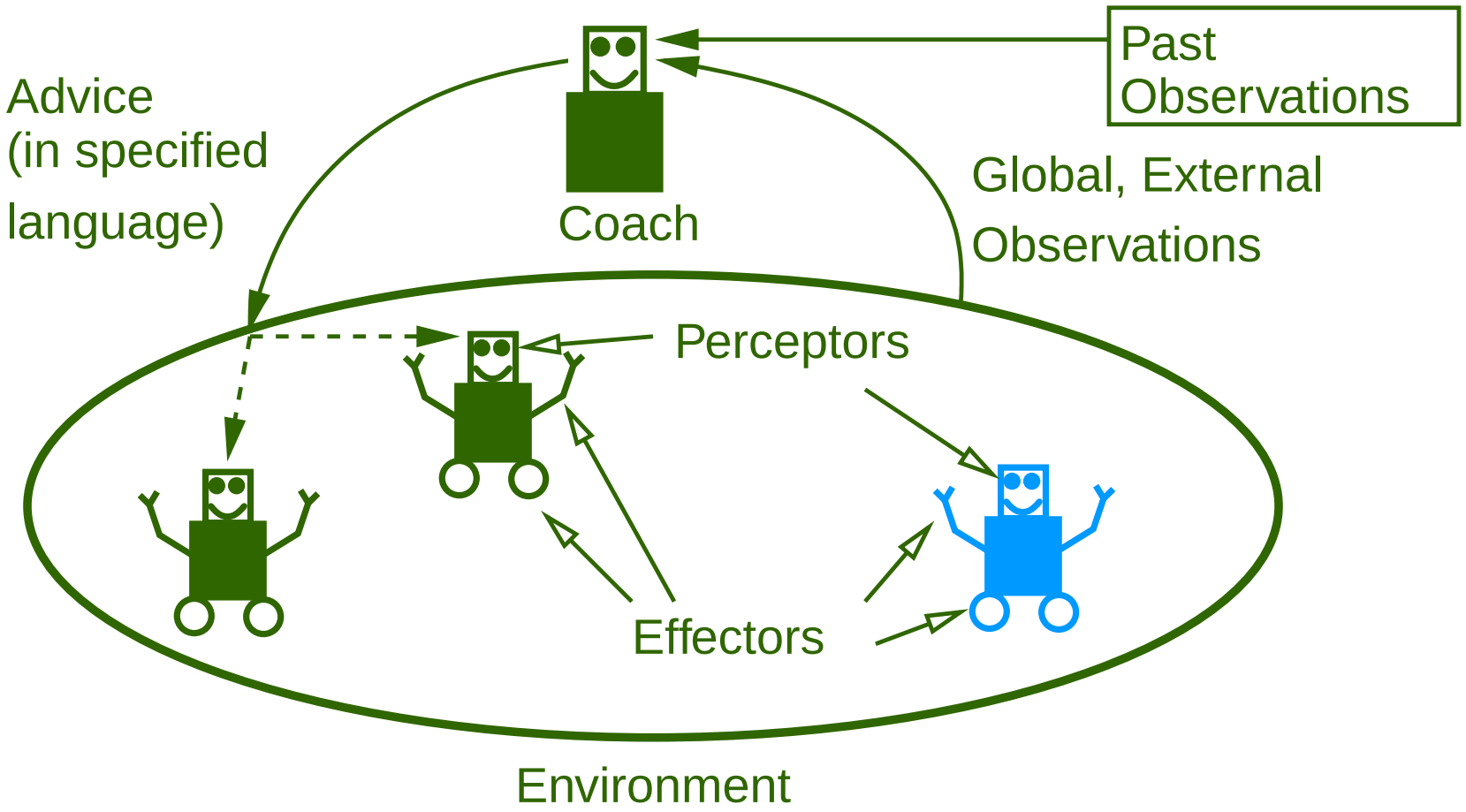
Coaching?



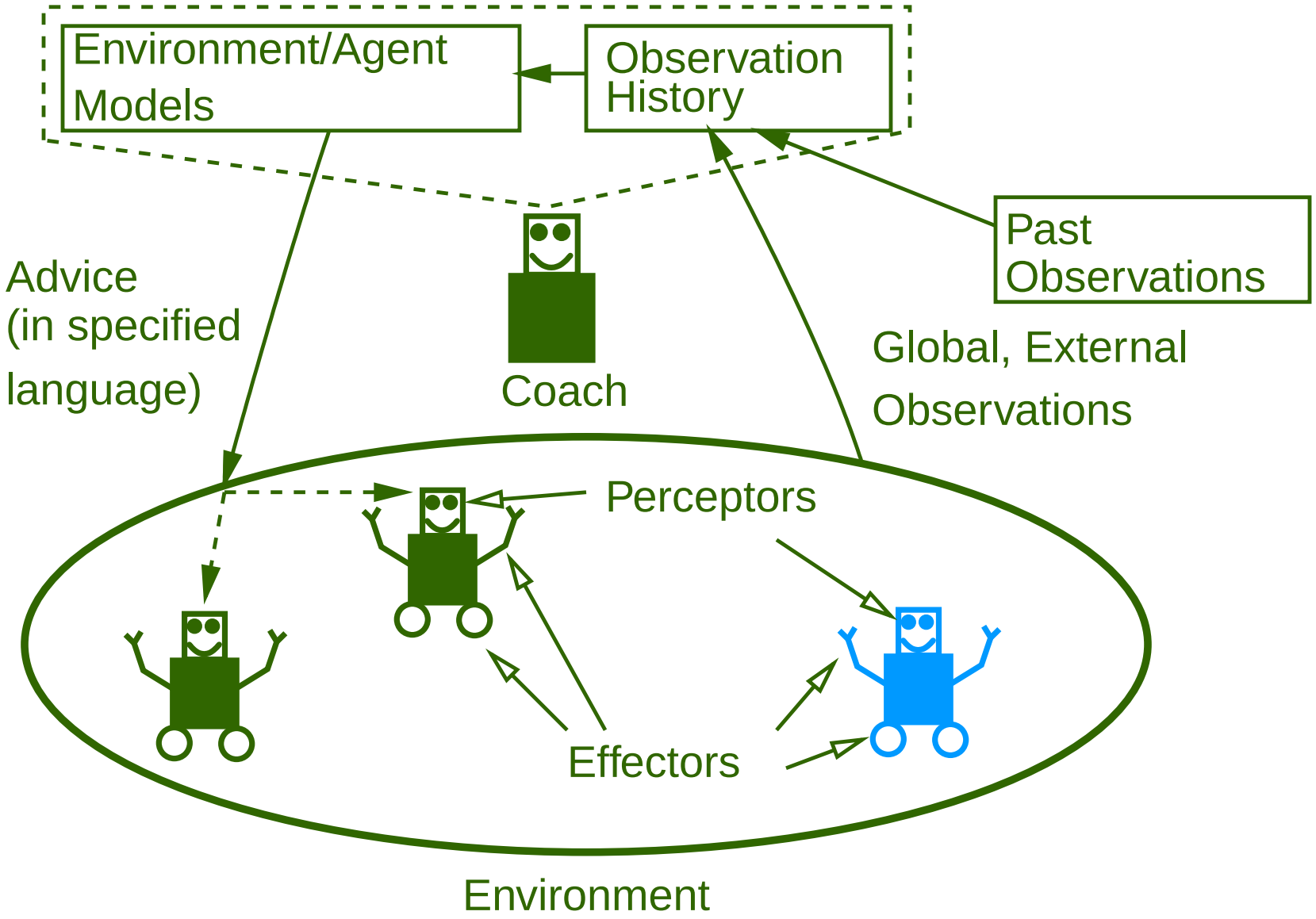
Coaching?


Advice
(in specified
language)
Coach
Global, External
Observations
Perceptors
Effectors
Environment

Coaching?


Past
Observations
Advice
(in specified
language)
Coach
Global, External
Observations
Perceptors
Effectors
Environment

Coaching?


Environment/Agent Models
Observation History
Past Observations
Advice (in specified language)
Coach
Global, External Observations
Perceptors
Effectors
Environment

Thesis Question

What algorithms can be used by an automated coach agent to provide advice to one or more agents in order to improve their performance?

Outline

- Prologue
  - Robot soccer environment
  - Coaching sub-questions

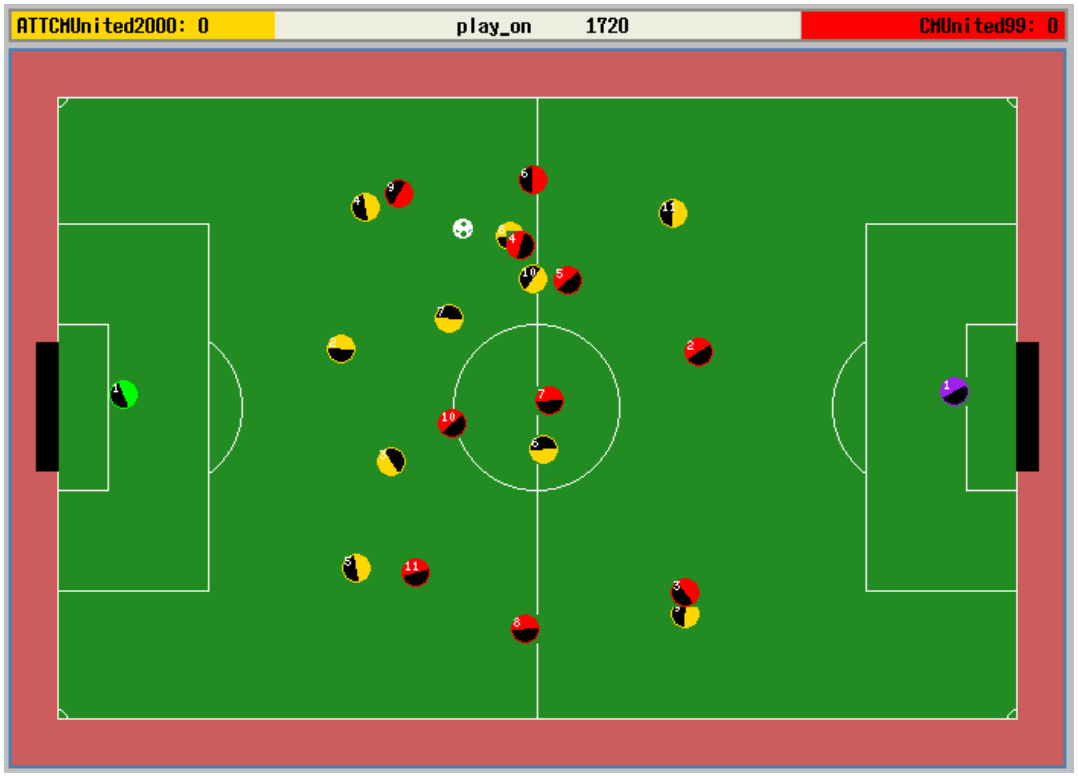
Outline

- Prologue
  - Robot soccer environment
  - Coaching sub-questions
- Technical sections
  - Matching opponents to models
  - Learning/using environment models

Outline

- Prologue
  - Robot soccer environment
  - Coaching sub-questions
- Technical sections
  - Matching opponents to models
  - Learning/using environment models
- Epilogue
  - Relation to previous work
  - Review/overview of thesis contributions
  - Future work

Motivating Environment: Simulated Robot Soccer



- Real time constraints
- Noisy actions
- Noisy and incomplete sensation
- Near continuous state/action spaces

- 22 distributed player agents

Simulated Robot Soccer: Coaching

- Coach agent with global view and limited communication
  - Coach does **not** see agent actions or intentions

Simulated Robot Soccer: Coaching

- Coach agent with global view and limited communication
  - Coach does **not** see agent actions or intentions
- Community created standard advice language named CLang
  - Rule based
  - Conditions are logical combinations of world state atoms
  - Actions are recommended macro-actions like passing and positioning

Simulated Robot Soccer: Coaching

- Coach agent with global view and limited communication
  - Coach does **not** see agent actions or intentions
- Community created standard advice language named CLang
  - Rule based
  - Conditions are logical combinations of world state atoms
  - Actions are recommended macro-actions like passing and positioning
- Basis for 4 years of coach competitions at RoboCup events
  - Run different coaches with same teams

My Questions in Coaching

- What can the coach learn from observations?
  - Opponent models; learn and/or select from given set
  - Learn environment models

My Questions in Coaching

- What can the coach learn from observations?
  - Opponent models; learn and/or select from given set
  - Learn environment models
- How can models be used to get desired actions for agents?
  - Plan a response to predicted behavior
  - Imitate a good team
  - Solve for universal plan

My Questions in Coaching

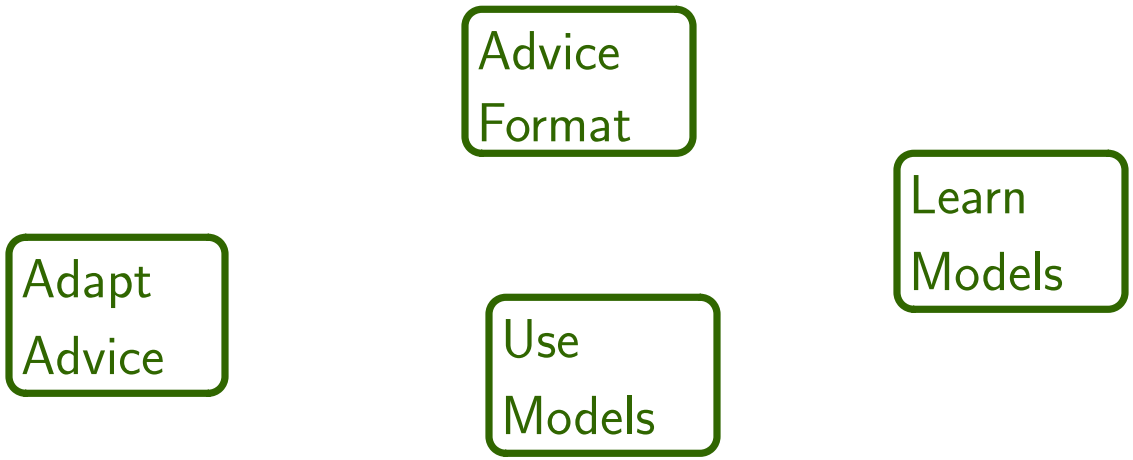
- What can the coach learn from observations?
  - Opponent models; learn and/or select from given set
  - Learn environment models
- How can models be used to get desired actions for agents?
  - Plan a response to predicted behavior
  - Imitate a good team
  - Solve for universal plan
- Once the coach has desired actions, how does the coach adapt advice to the agent abilities?

My Questions in Coaching

- What can the coach learn from observations?
  - Opponent models; learn and/or select from given set
  - Learn environment models
- How can models be used to get desired actions for agents?
  - Plan a response to predicted behavior
  - Imitate a good team
  - Solve for universal plan
- Once the coach has desired actions, how does the coach adapt advice to the agent abilities?
- What format does advice take?

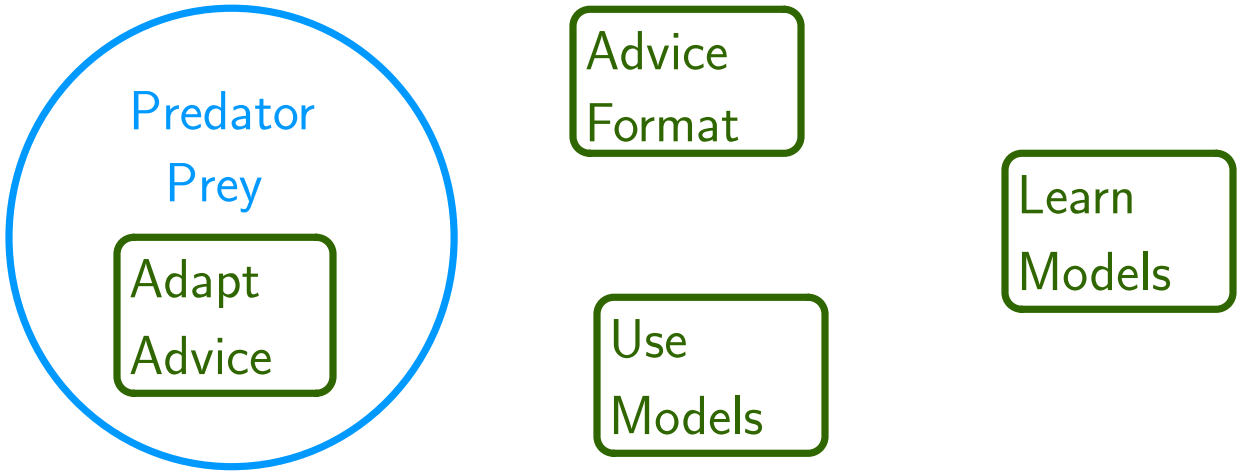
How to Study Coaching?

- Isolate questions with various domains



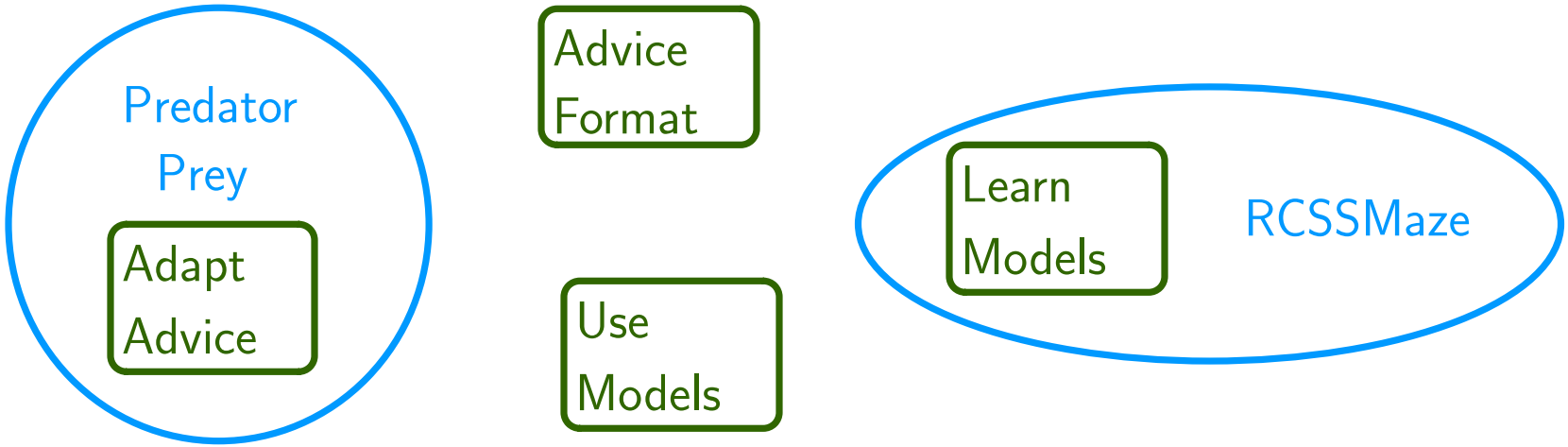
How to Study Coaching?

- Isolate questions with various domains



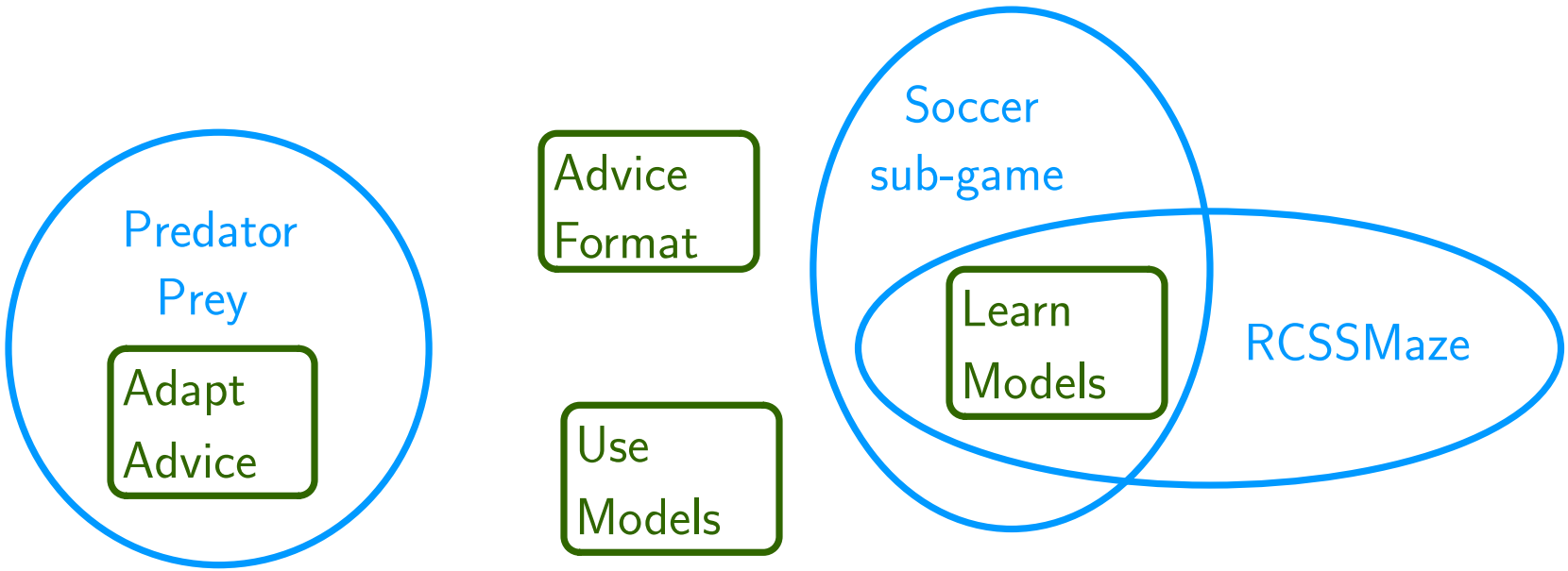
How to Study Coaching?

- Isolate questions with various domains



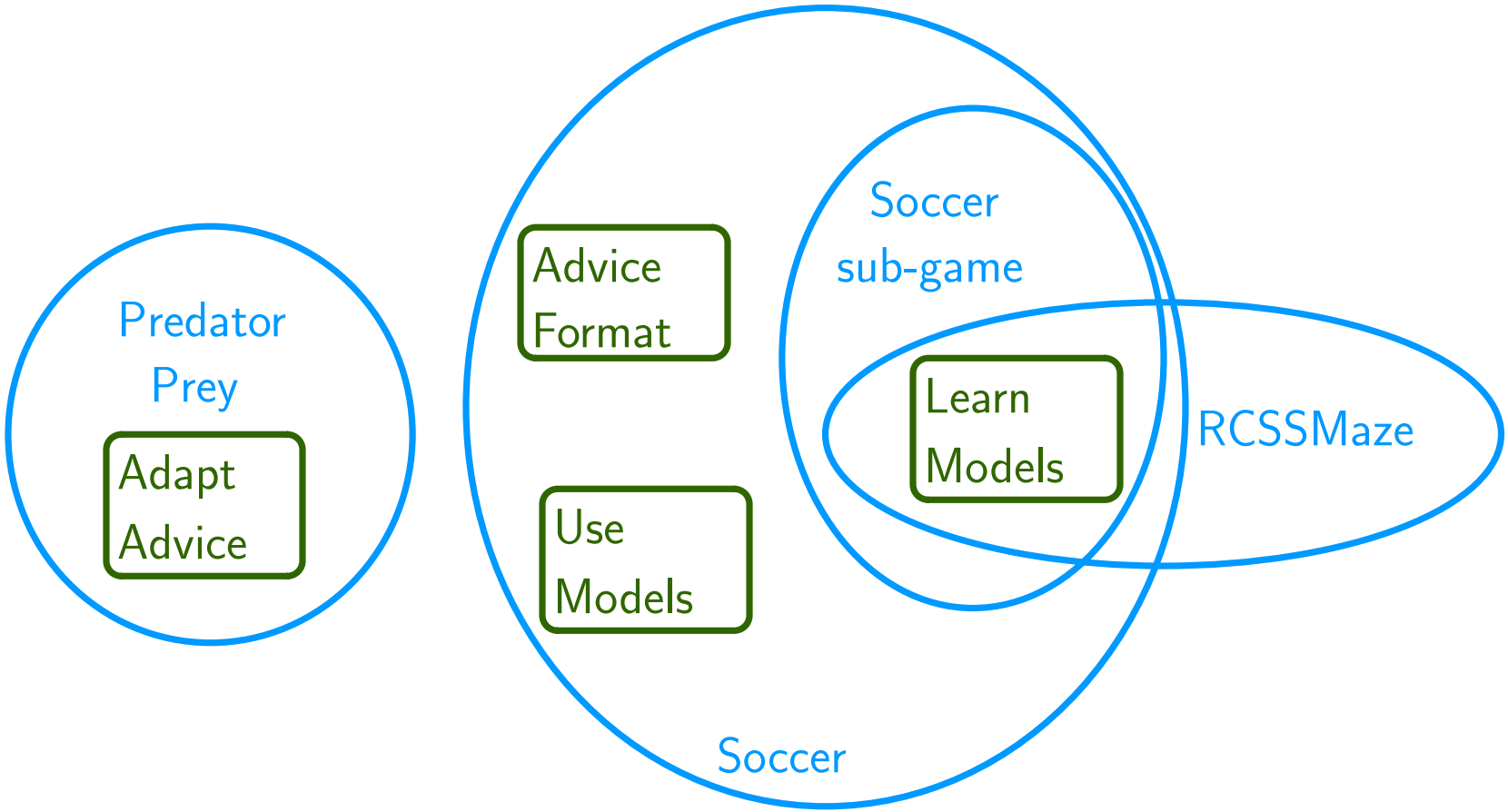
How to Study Coaching?

- Isolate questions with various domains



How to Study Coaching?

- Isolate questions with various domains



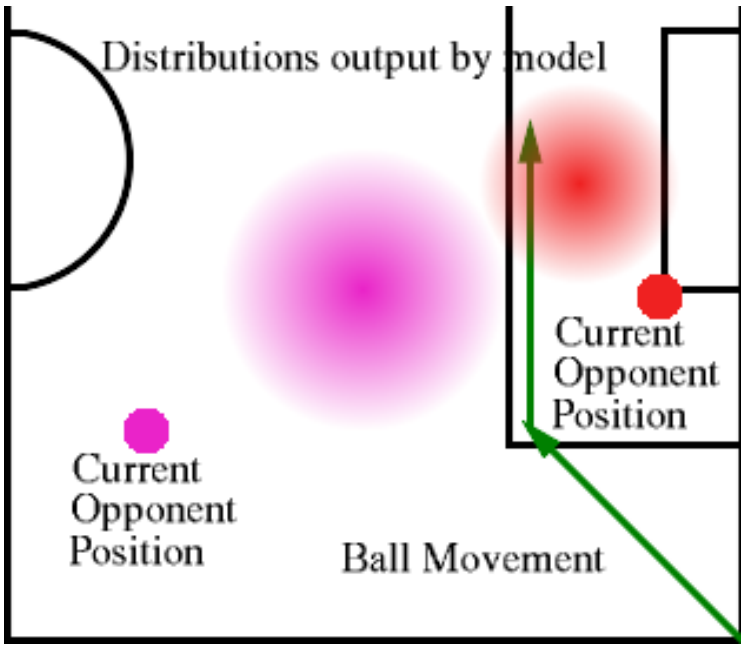
Opponent Models

Why Opponent Models?

- Dealing with opponents is a fertile area for advice
- Adapting to current opponent can mean better performance

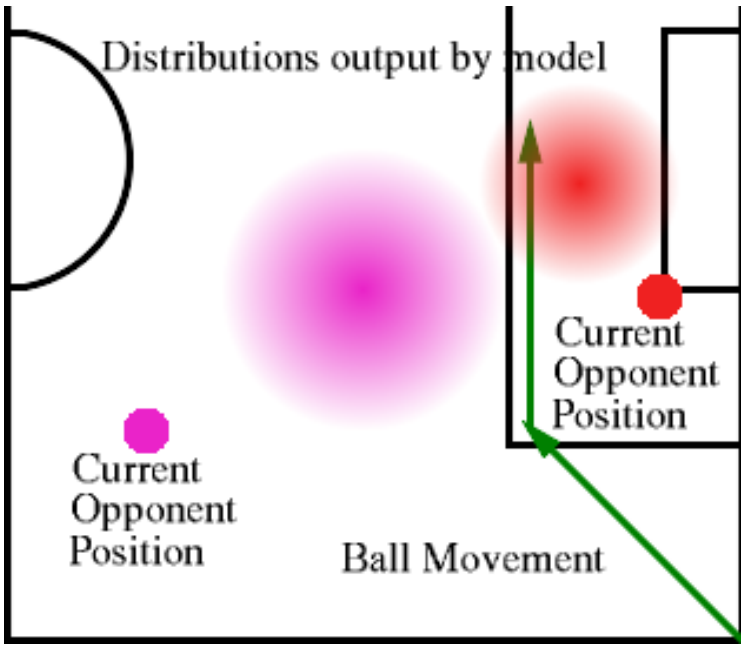
Predicting Opponent Movement

Predicting Opponent Movement



Predicting Opponent Movement

$$M: \mathcal{S}_W \times \mathcal{S}_O^p \times \mathcal{A} \rightarrow \mathcal{R}_O^p$$



M Opponent model

$\mathcal{S}_W$ Set of world states

p Players per team

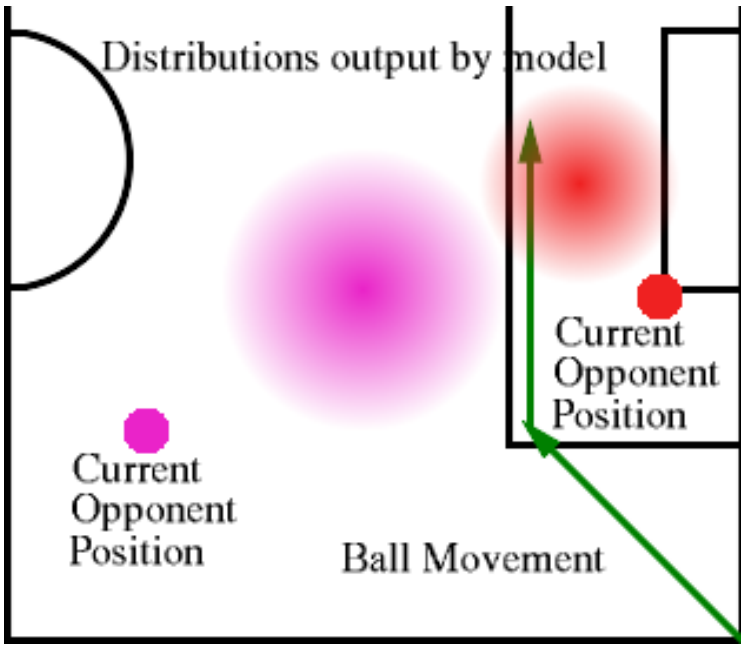
$\mathcal{S}_O$ Set of opponent states

$\mathcal{A}$ Planned actions of our team

$\mathcal{R}_O$ Probability distribution over opponent states

Predicting Opponent Movement

$$M: \mathcal{S}_W \times \mathcal{S}_O^p \times \mathcal{A} \rightarrow \mathcal{R}_O^p$$



M Opponent model

$\mathcal{S}_W$ Set of world states

p Players per team

$\mathcal{S}_O$ Set of opponent states

$\mathcal{A}$ Planned actions of our team

$\mathcal{R}_O$ Probability distribution over opponent states

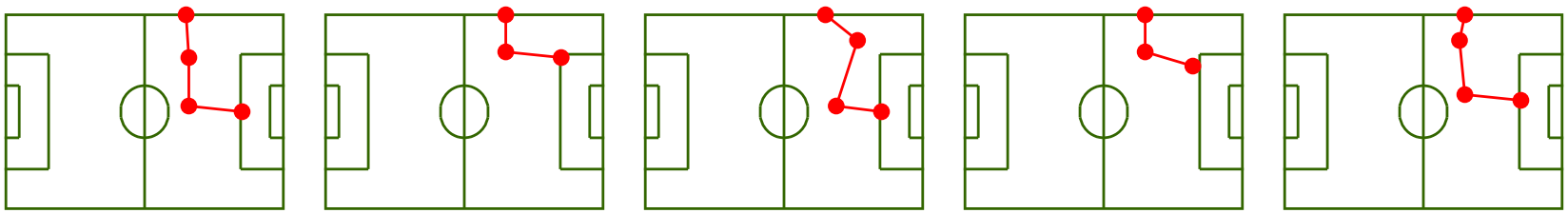
- Use predicted opponent movement to plan team actions

Selecting Between Opponent Models

- Online, must make quick decisions with small amounts of data
- Rather than learning a new model from scratch, coach will select between models from a predefined set

Selecting Between Opponent Models

- Online, must make quick decisions with small amounts of data
- Rather than learning a new model from scratch, coach will select between models from a predefined set
- Model chosen affects the plan generated



Model 1 Model 2 Model 3 Model 4 Model 5

Selecting Between Opponent Models

- Maintain probability distribution over set of models $P[M_i]$
- Use observation $o = (w, s, a, e)$ to update with naive Bayes

  w World state (ball location)
  s Starting opponent states (locations)
  a Team actions (ball movement)
  e Ending opponent states (locations)

Selecting Between Opponent Models

- Maintain probability distribution over set of models $P[M_i]$
- Use observation $o = (w, s, a, e)$ to update with naive Bayes

  w World state (ball location)
  s Starting opponent states (locations)
  a Team actions (ball movement)
  e Ending opponent states (locations)

$$P[M_i|o] = \underbrace{P[e_1|w,s,a,M_i]P[e_2|w,s,a,M_i]\ldots P[e_p|w,s,a,M_i]}_{\text{what opponent model calculates}} \underbrace{\frac{P[w,s,a]}{P[o]}}_{\text{norm. constant}} \underbrace{P[M_i]}_{\text{prior}}$$

Can Models Be Recognized?

We presented an algorithm to select a model from a set. Does it select the correct one?

Can Models Be Recognized?

We presented an algorithm to select a model from a set. Does it select the correct one?

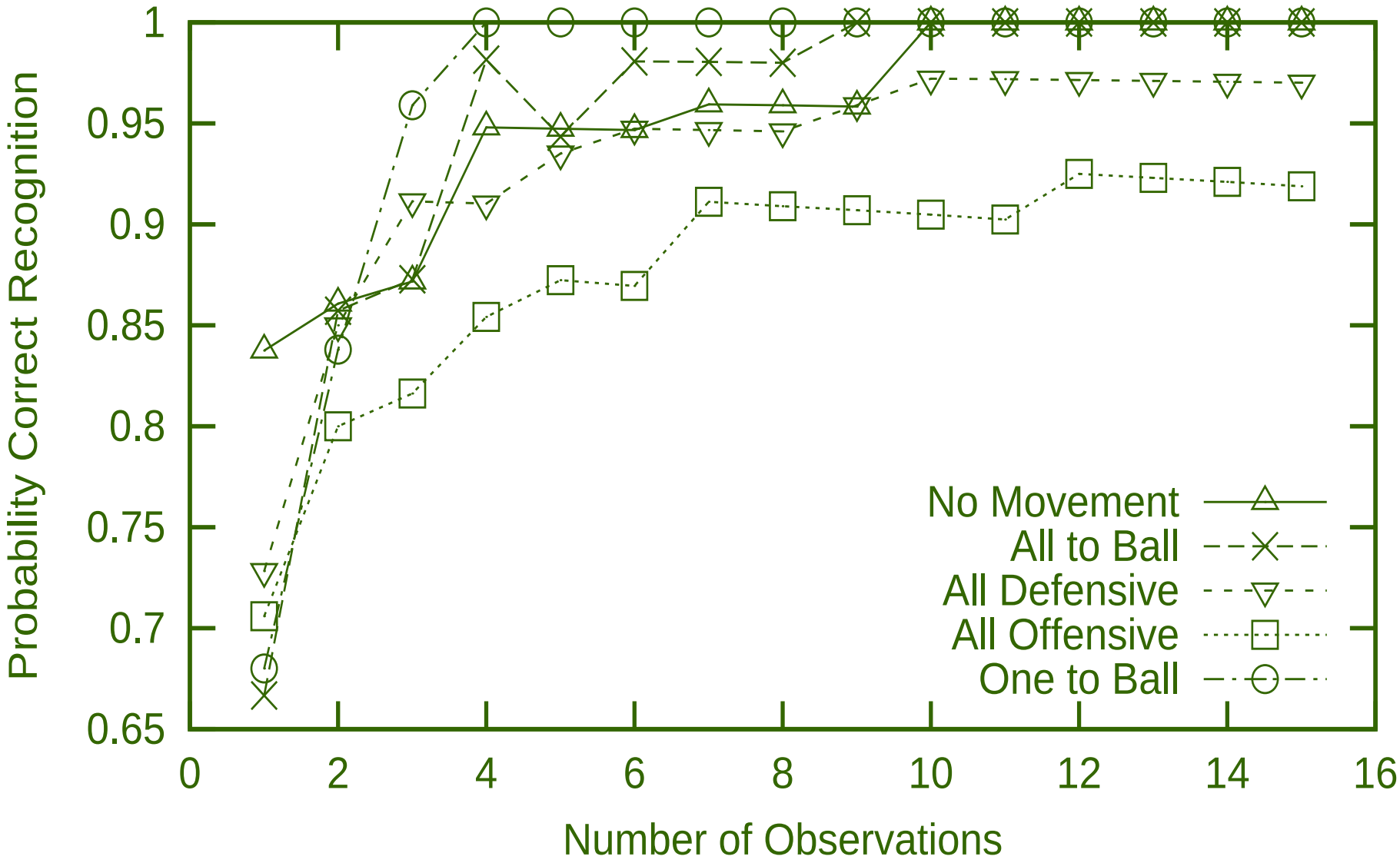
- Define a set of five models
- Define a set of teams that (mostly) act like the models

Can Models Be Recognized?

We presented an algorithm to select a model from a set. Does it select the correct one?

- Define a set of five models
- Define a set of teams that (mostly) act like the models
- Observe each of the five teams playing while the coach makes plans
- For each of the teams, how often is the correct model selected?

Recognition Results



Environment Models

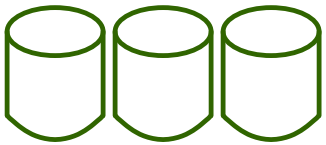
Environment Model?

- Model the effects of possible agent actions on the state of the world
  - Our algorithms learn an abstract Markov Decision Process

Environment Model?

- Model the effects of possible agent actions on the state of the world
  - Our algorithms learn an abstract Markov Decision Process
- A coach must have some knowledge to provide advice
- An environment model can be solved to get a desired action policy for the agents

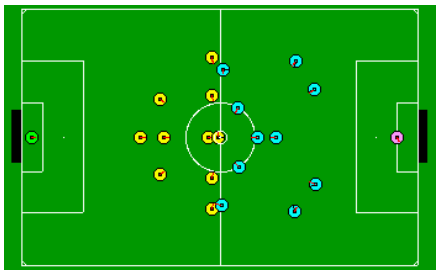
Observations, ..., Advice

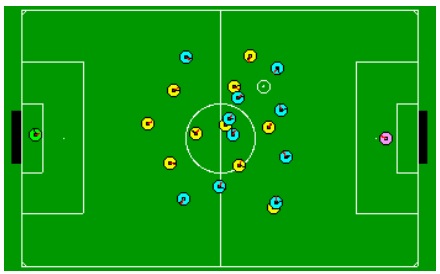


Observations of Past Execution

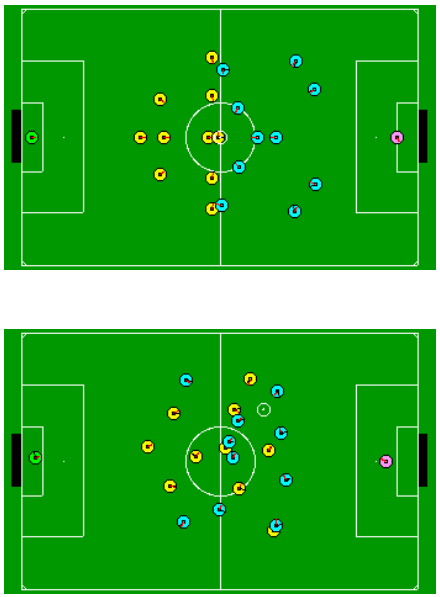
Advice

What are Observations?





What are Observations?



t, score, play mode,

$\langle x_{\text{ball}}, y_{\text{ball}}, \Delta x_{\text{ball}}, \Delta y_{\text{ball}} \rangle$

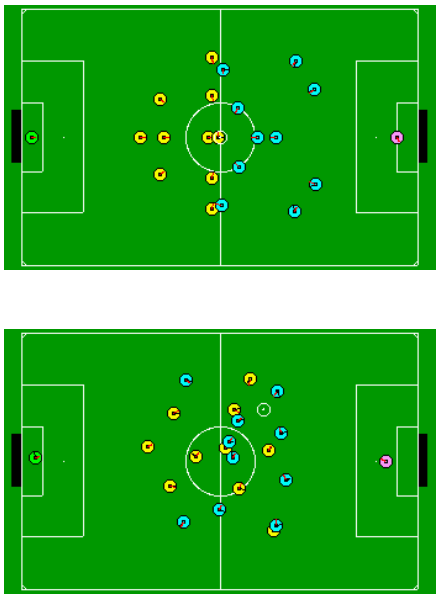
$\langle x_1, y_1, \Delta x_1 \Delta y_1, \theta_1^B, \theta_1^N, \text{view}_1, \ldots \rangle$

$\langle x_2, y_2, \Delta x_2 \Delta y_2, \theta_2^B, \theta_2^N, \text{view}_2, \ldots \rangle$

$\vdots$

$\langle x_{22}, y_{22}, \Delta x_{22} \Delta y_{22}, \theta_{22}^B, \theta_{22}^N, \text{view}_{22}, \ldots \rangle$

What are Observations?



t, score, play mode,

$\langle x_{\text{ball}}, y_{\text{ball}}, \Delta x_{\text{ball}}, \Delta y_{\text{ball}} \rangle$

$\langle x_1, y_1, \Delta x_1 \Delta y_1, \theta_1^B, \theta_1^N, \text{view}_1, \ldots \rangle$

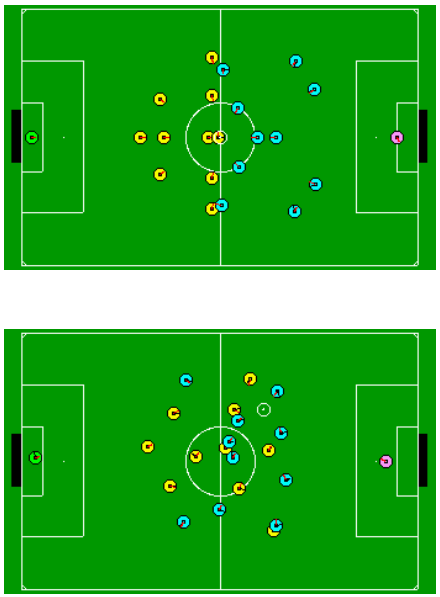
$\langle x_2, y_2, \Delta x_2 \Delta y_2, \theta_2^B, \theta_2^N, \text{view}_2, \ldots \rangle$

$\vdots$

$\langle x_{22}, y_{22}, \Delta x_{22} \Delta y_{22}, \theta_{22}^B, \theta_{22}^N, \text{view}_{22}, \ldots \rangle$

- Only state, no actions
  - But produced by agents taking actions
- Externally visible global view

What are Observations?



t, score, play mode,

$\langle x_{\text{ball}}, y_{\text{ball}}, \Delta x_{\text{ball}}, \Delta y_{\text{ball}} \rangle$

$\langle x_1, y_1, \Delta x_1 \Delta y_1, \theta_1^B, \theta_1^N, \text{view}_1, \ldots \rangle$

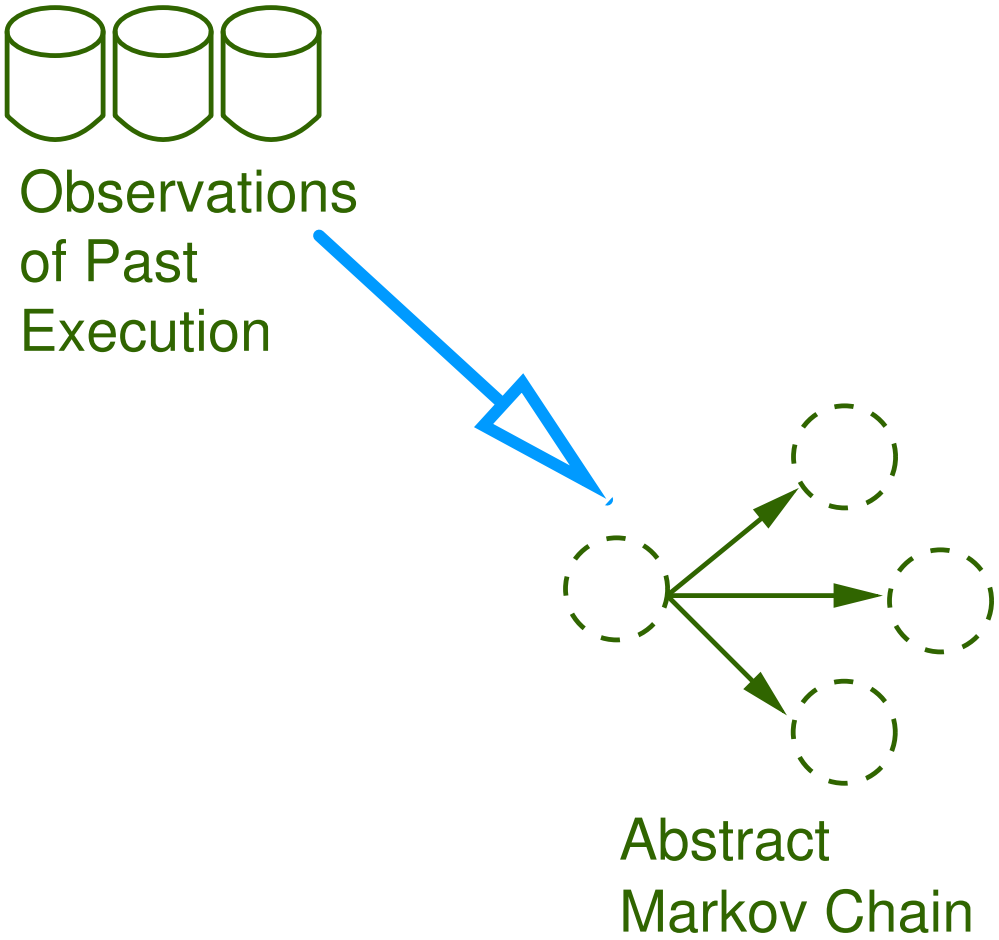
$\langle x_2, y_2, \Delta x_2 \Delta y_2, \theta_2^B, \theta_2^N, \text{view}_2, \ldots \rangle$

$\vdots$

$\langle x_{22}, y_{22}, \Delta x_{22} \Delta y_{22}, \theta_{22}^B, \theta_{22}^N, \text{view}_{22}, \ldots \rangle$

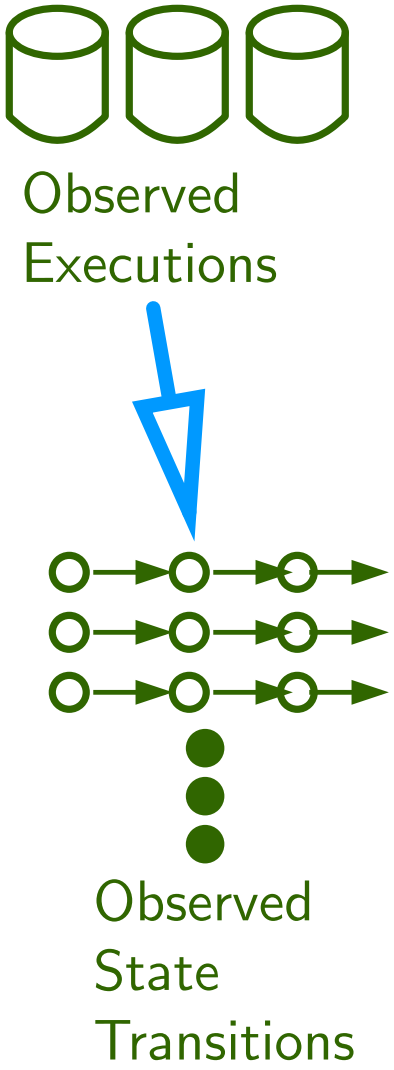
- Only state, no actions
  - But produced by agents taking actions
- Externally visible global view
- Observation logs exists for many processes, not just soccer

Observations, Markov Chain, ..., Advice

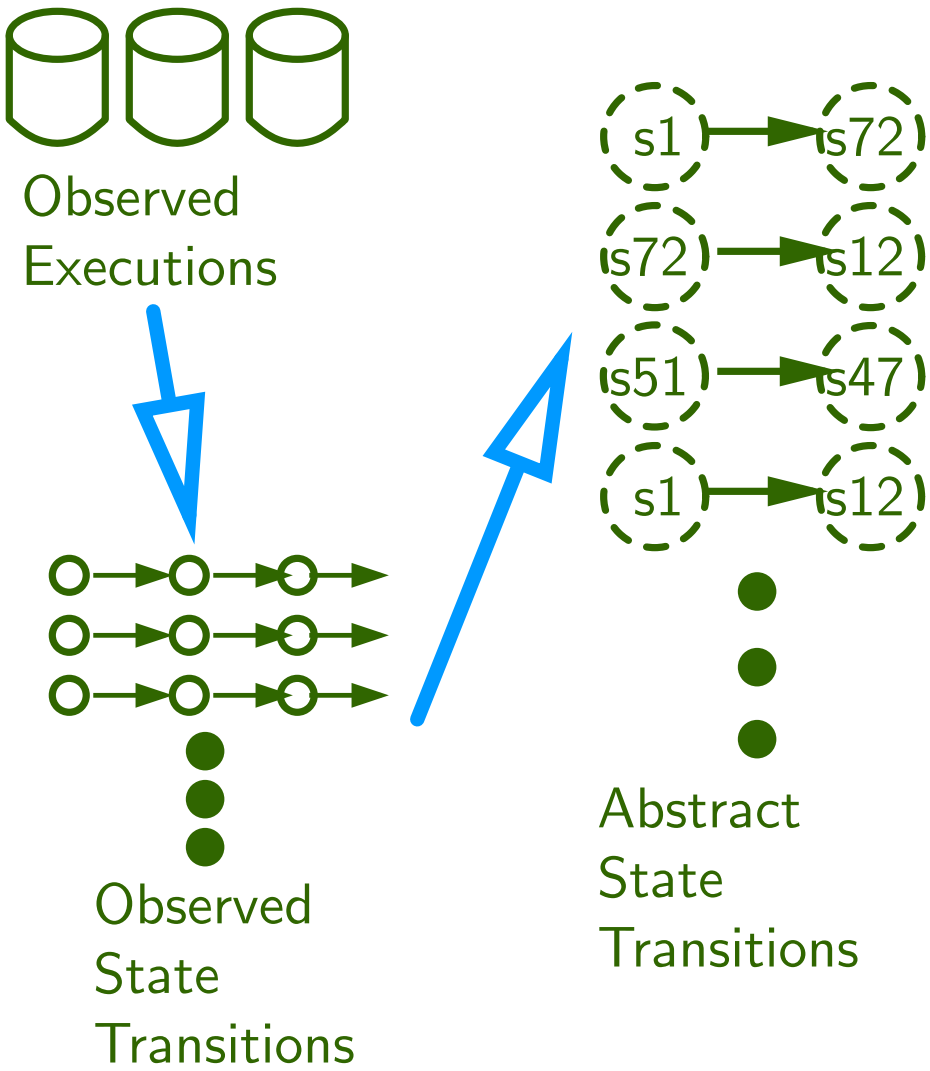


Advice

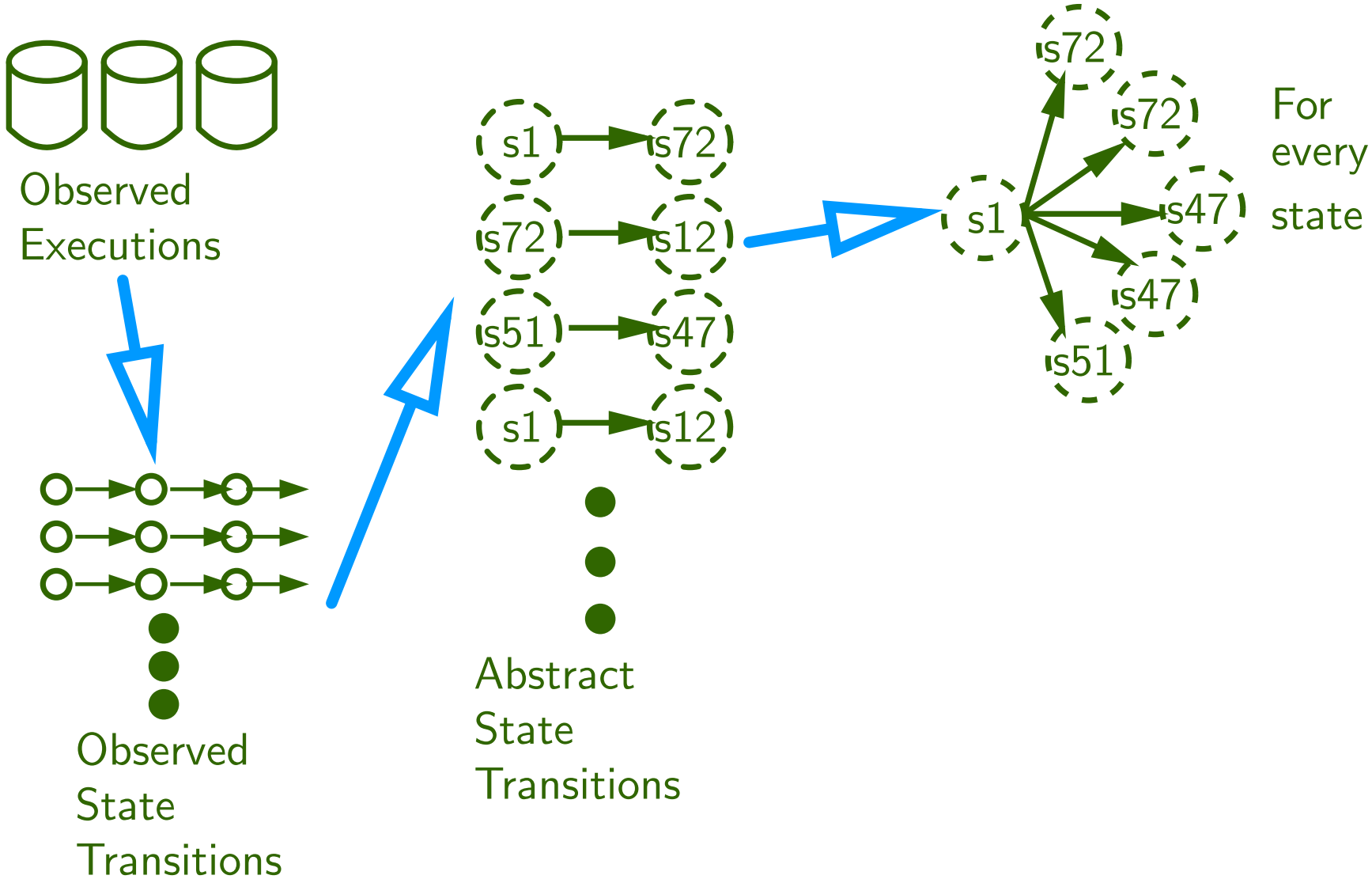
Observations to Markov Chain



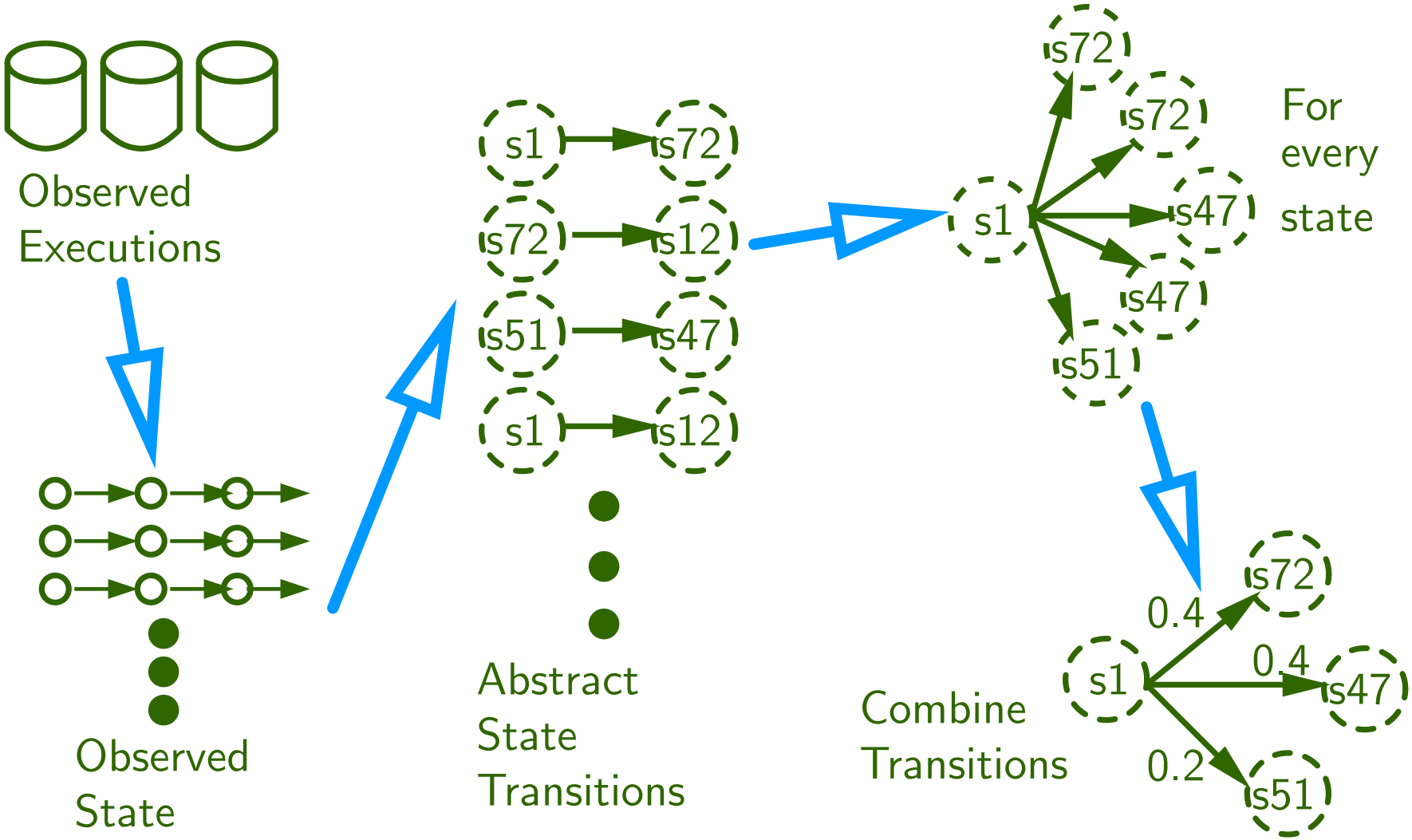
Observations to Markov Chain



Observations to Markov Chain

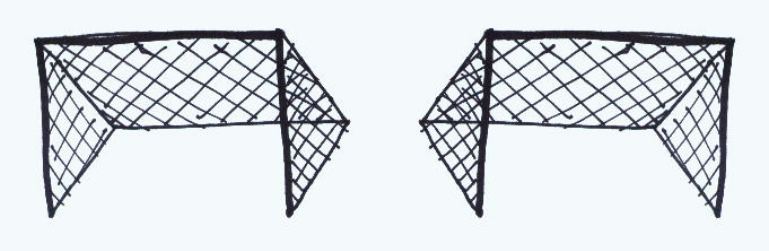


Observations to Markov Chain



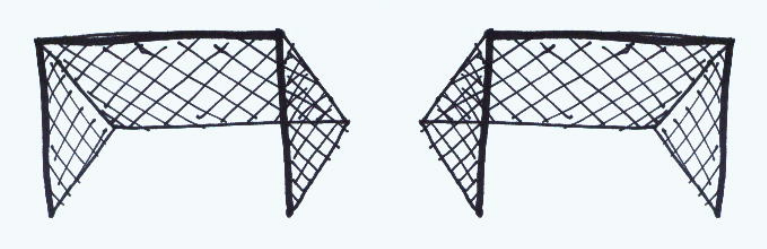
State Abstraction in Robot Soccer

Goal



State Abstraction in Robot Soccer

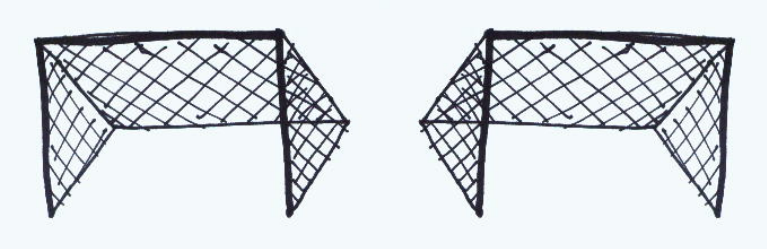
Goal



Ball possession

State Abstraction in Robot Soccer

Goal



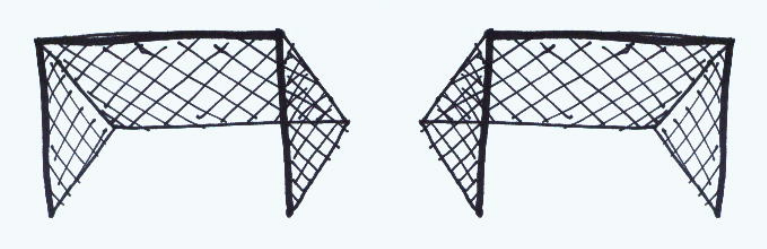
Ball possession

Ball grid

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59

State Abstraction in Robot Soccer

Goal

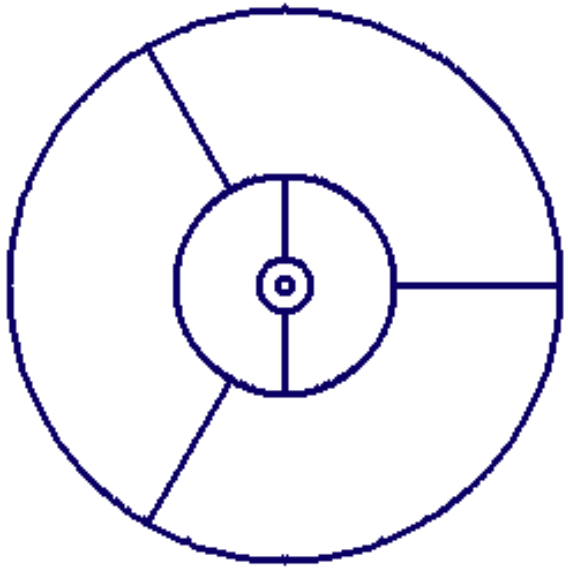


Ball possession

Ball grid

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59

Player occupancy



Observations to Markov Chain: Formalism

$s'_3 \rightarrow s'_9 \rightarrow s'_3 \rightarrow s'_2 \ldots$
$s'_9 \rightarrow s'_3 \rightarrow s'_2 \rightarrow s'_7 \ldots$
$s'_i \in S$

Observation Data

Abstract State

$\langle \bar{S}, B \colon S \rightarrow \bar{S} \cup \varepsilon \rangle$

$\mathcal{S}$ Set of observation states

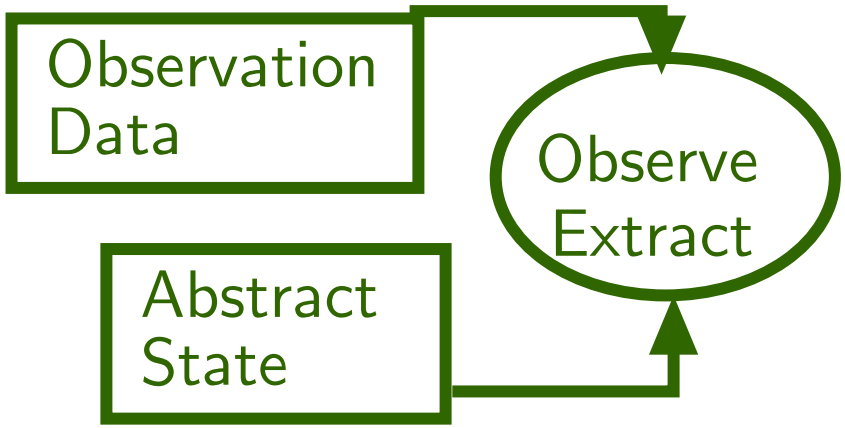
$\bar{\mathcal{S}}$ Set of abstract states

B Abstraction function

T_{MC} Transition function

Observations to Markov Chain: Formalism

$s'_3 \rightarrow s'_9 \rightarrow s'_3 \rightarrow s'_2 \ldots$
$s'_9 \rightarrow s'_3 \rightarrow s'_2 \rightarrow s'_7 \ldots$
$s'_i \in S$



$\langle \bar{S}, B \colon S \rightarrow \bar{S} \cup \varepsilon \rangle$

$\mathcal{S}$ Set of observation states

$\bar{\mathcal{S}}$ Set of abstract states

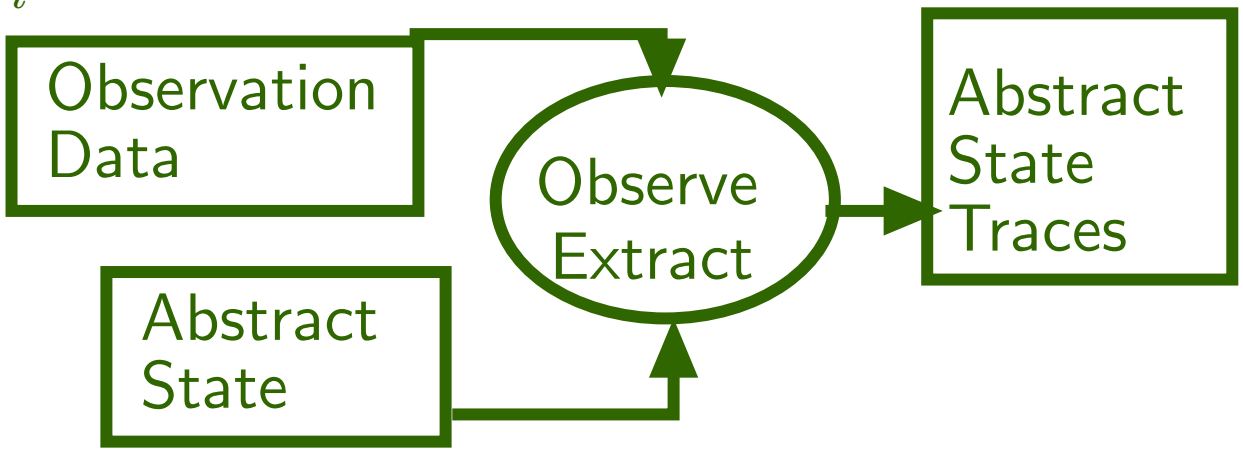
B Abstraction function

T_{MC} Transition function

Observations to Markov Chain: Formalism

$s'_3 \rightarrow s'_9 \rightarrow s'_3 \rightarrow s'_2 \ldots$
$s'_9 \rightarrow s'_3 \rightarrow s'_2 \rightarrow s'_7 \ldots$
$s'_i \in S$

$s_1 \rightarrow s_2 \rightarrow s_1 \ldots$
$s_2 \rightarrow s_1 \rightarrow s_2 \ldots$
$s_i \in \bar{S}$



$\langle \bar{S}, B \colon S \rightarrow \bar{S} \cup \varepsilon \rangle$

$\mathcal{S}$ Set of observation states

$\bar{\mathcal{S}}$ Set of abstract states

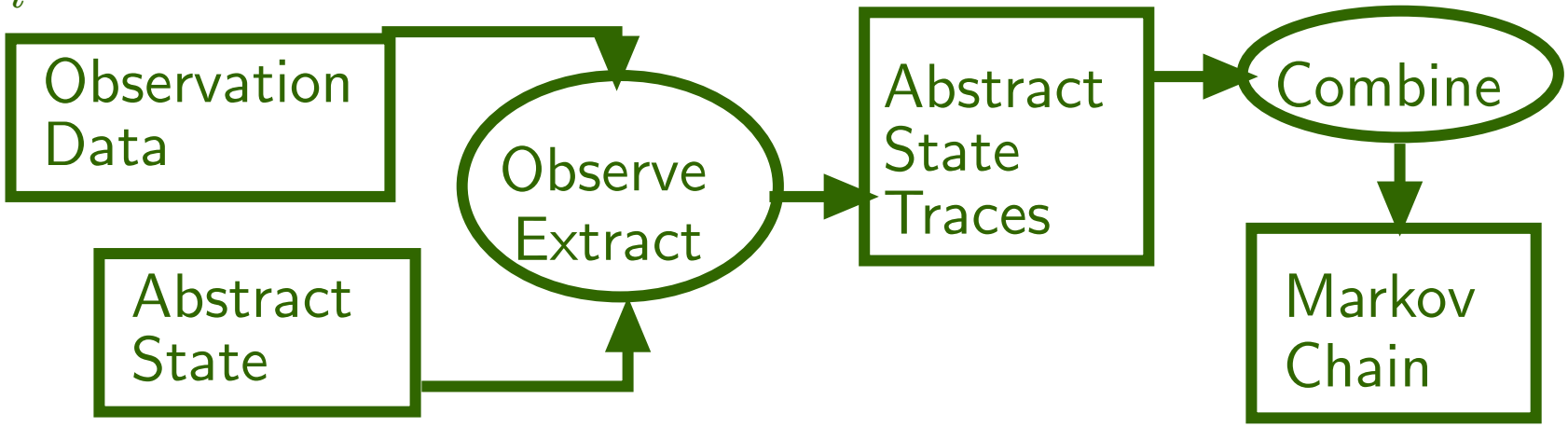
B Abstraction function

T_{MC} Transition function

Observations to Markov Chain: Formalism

$s'_3 \rightarrow s'_9 \rightarrow s'_3 \rightarrow s'_2 \ldots$
$s'_9 \rightarrow s'_3 \rightarrow s'_2 \rightarrow s'_7 \ldots$
$s'_i \in S$

$s_1 \rightarrow s_2 \rightarrow s_1 \ldots$
$s_2 \rightarrow s_1 \rightarrow s_2 \ldots$
$s_i \in \bar{S}$



$\langle \bar{S}, B \colon S \rightarrow \bar{S} \cup \varepsilon \rangle$

$\langle \bar{S}, T_{MC} \rangle$

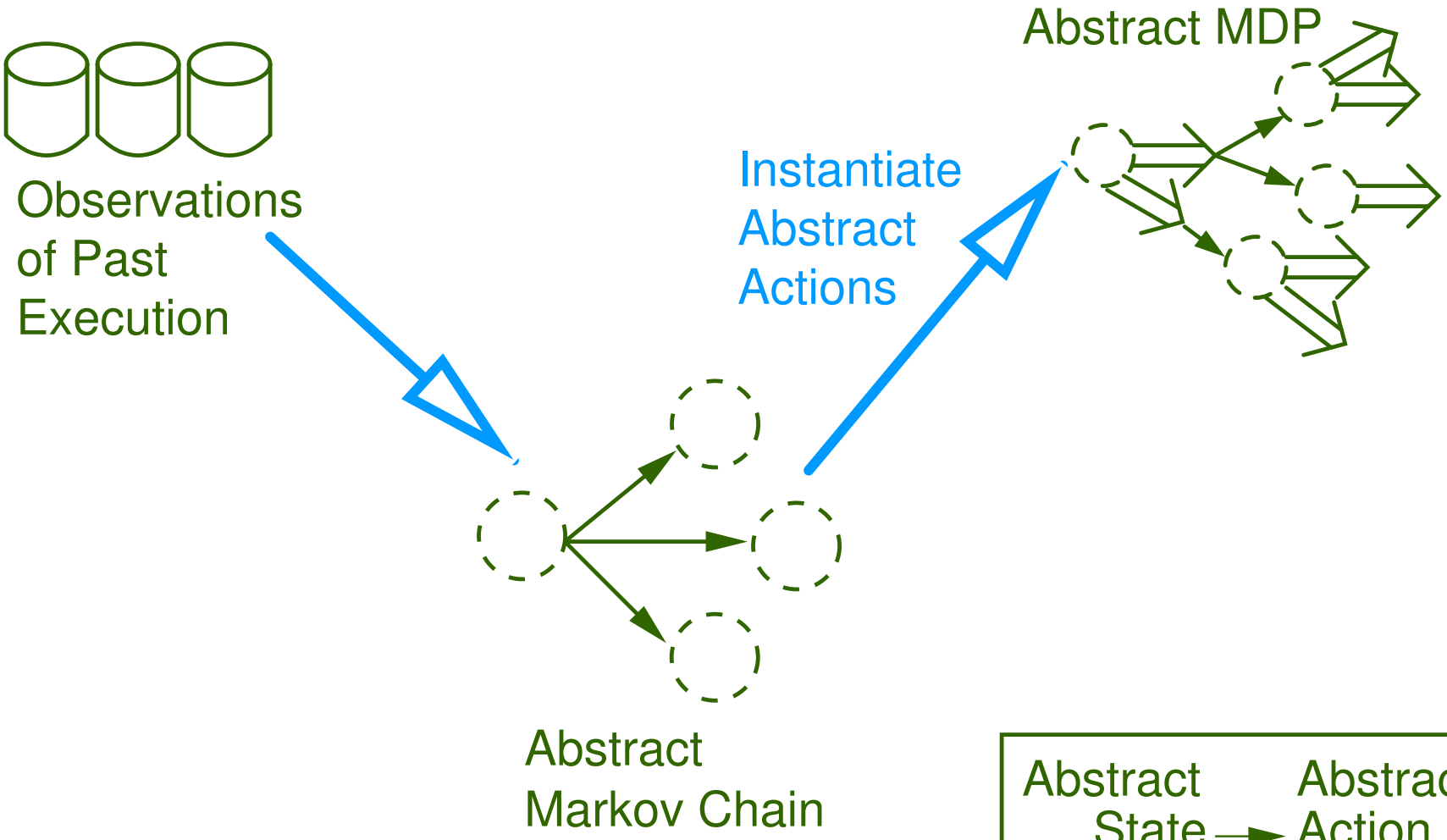
$\mathcal{S}$ Set of observation states

$\bar{\mathcal{S}}$ Set of abstract states

B Abstraction function

T_{MC} Transition function

Observations, MC, MDP, …, Advice



Markov Chain to MDP

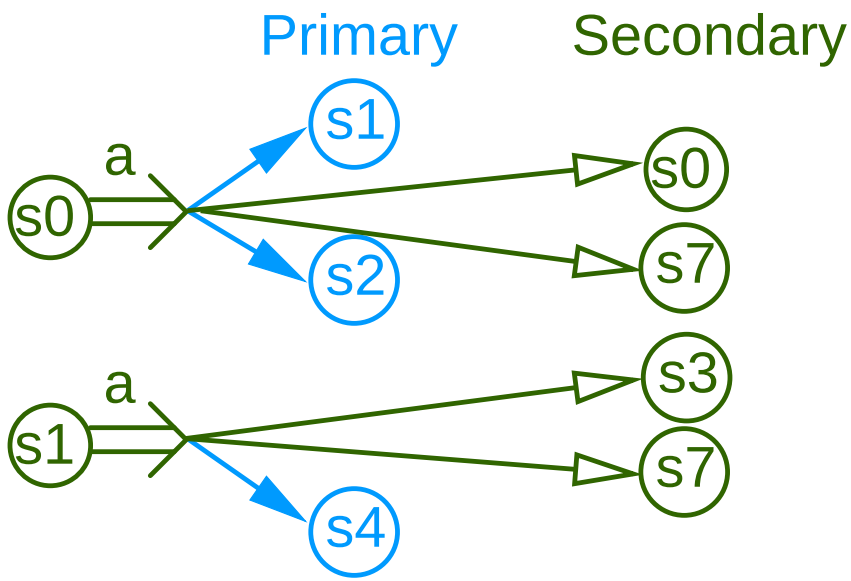
How to infer actions from Markov Chain?

- Solution: Introduce abstract action templates
  - Sets of primary and secondary transitions
  - Non-deterministic, but no probabilities

Markov Chain to MDP

How to infer actions from Markov Chain?

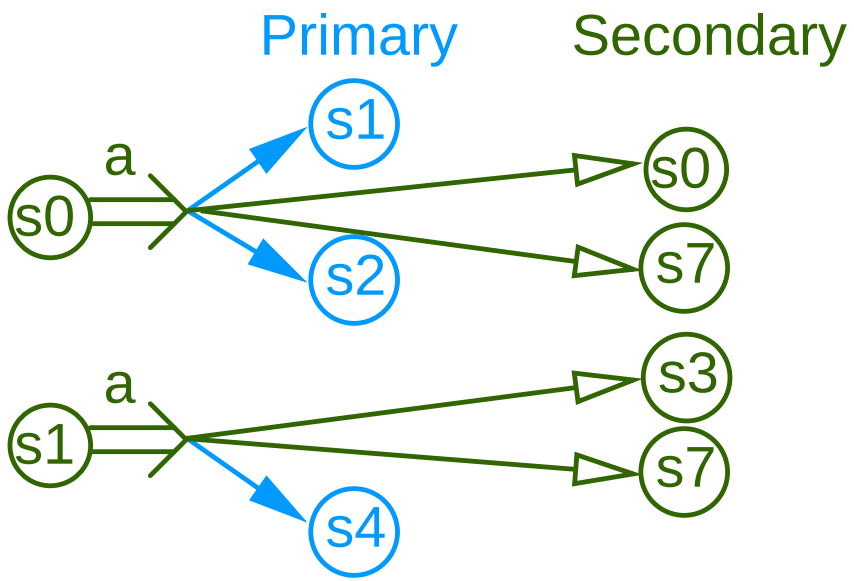
- Solution: Introduce abstract action templates
  - Sets of primary and secondary transitions
  - Non-deterministic, but no probabilities



Markov Chain to MDP

How to infer actions from Markov Chain?

- Solution: Introduce abstract action templates
  - Sets of primary and secondary transitions
  - Non-deterministic, but no probabilities

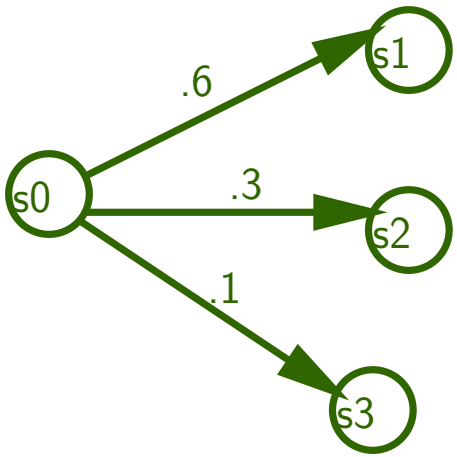


- Same action templates for different agents

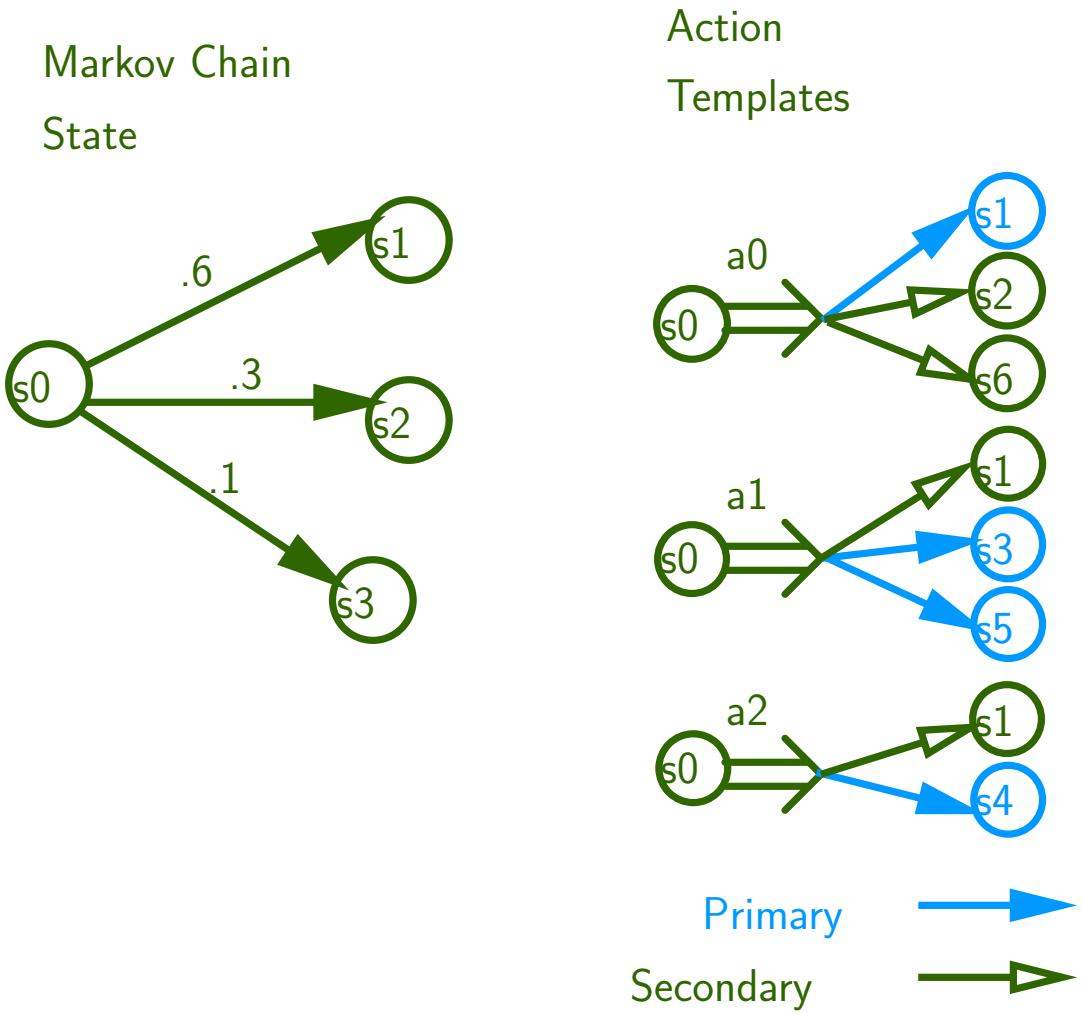
Markov Chain to MDP: Example

Markov Chain

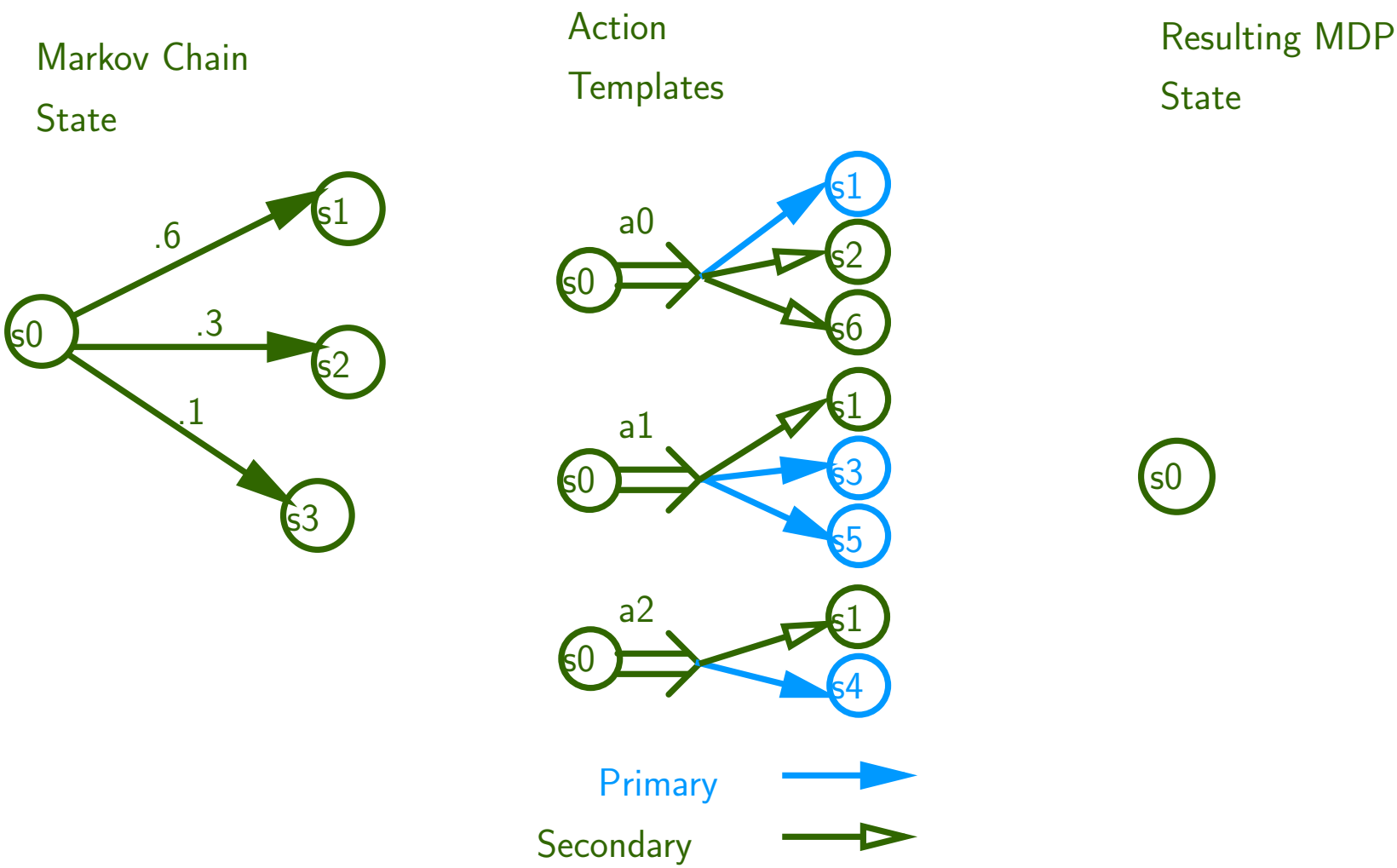
State



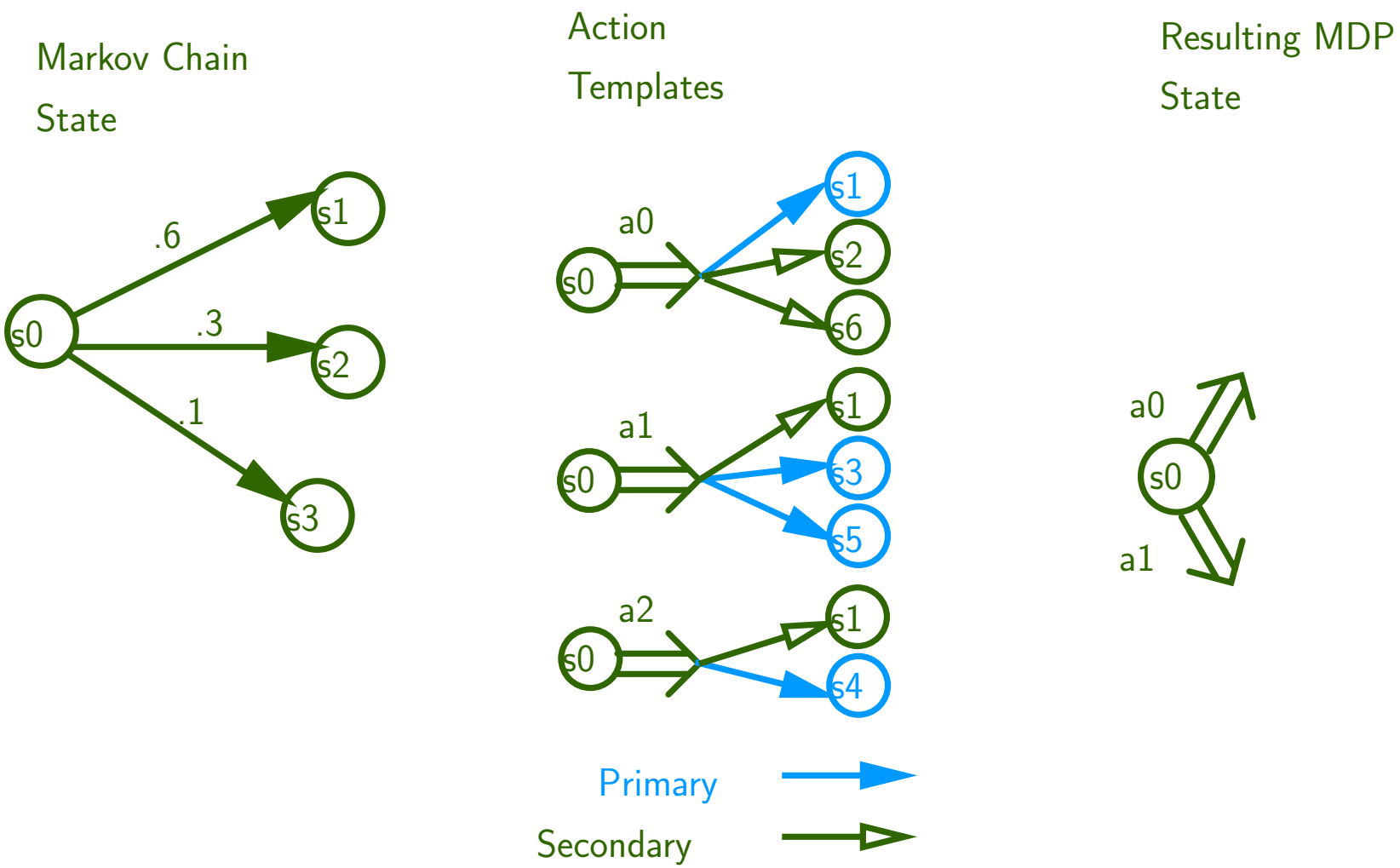
Markov Chain to MDP: Example



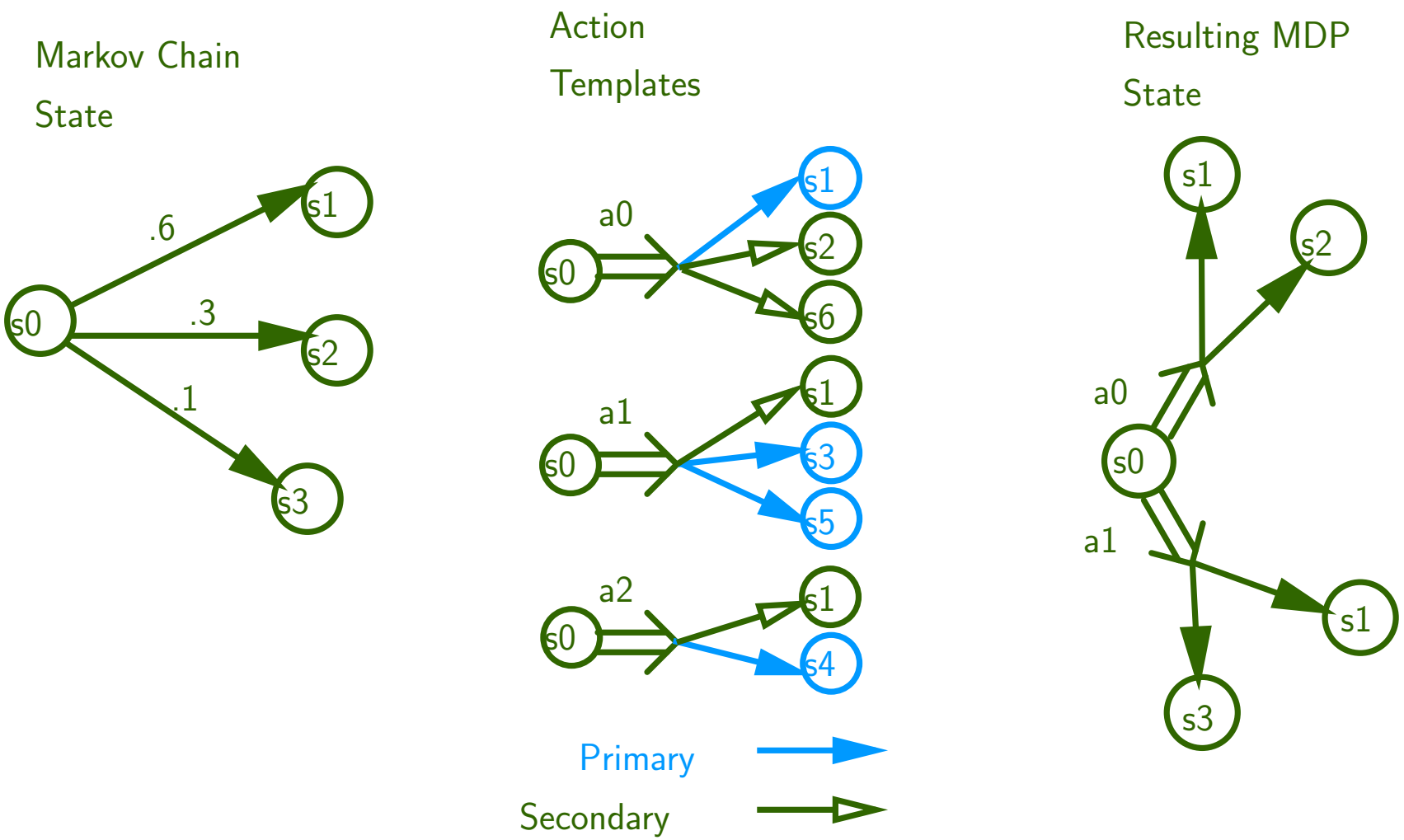
Markov Chain to MDP: Example



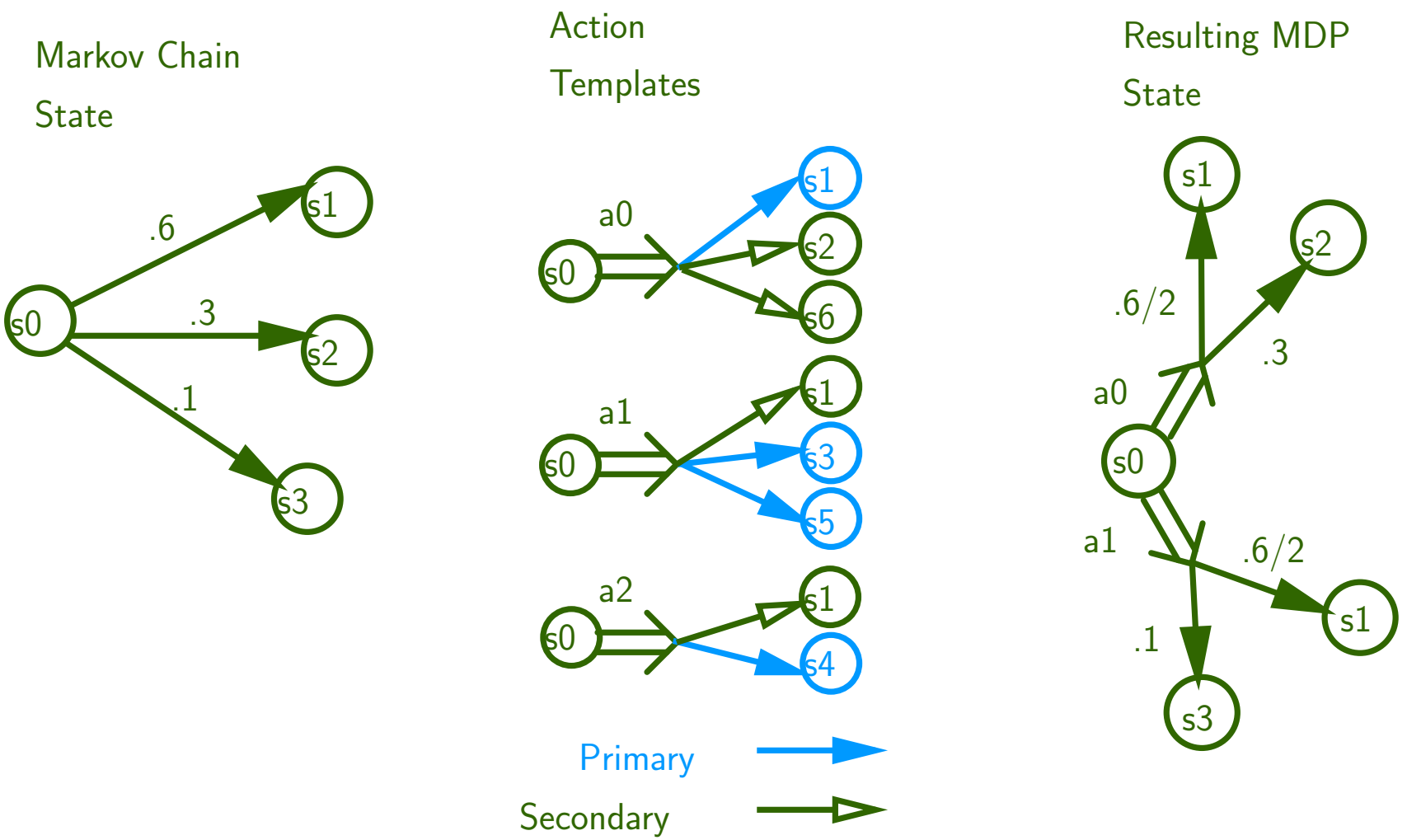
Markov Chain to MDP: Example



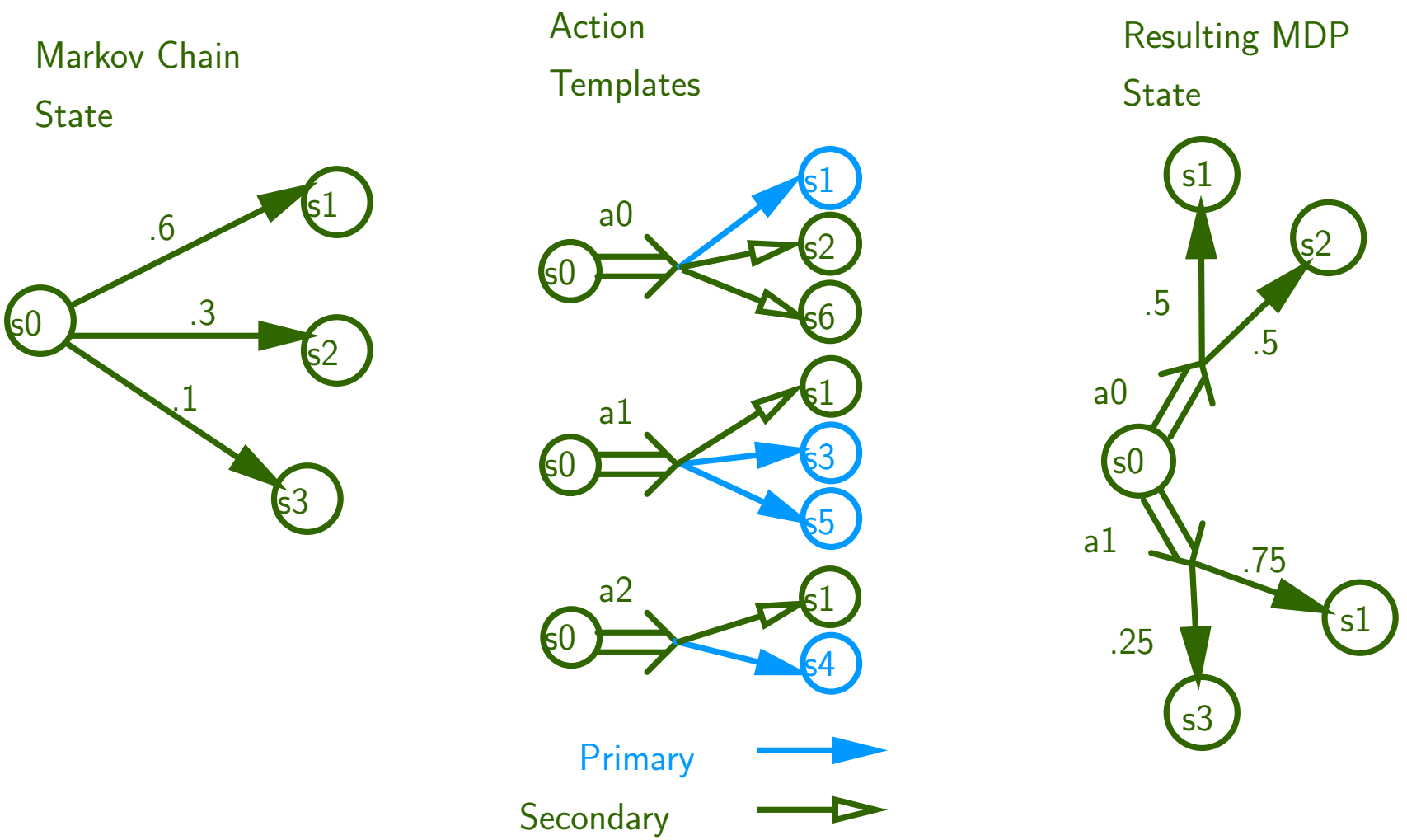
Markov Chain to MDP: Example



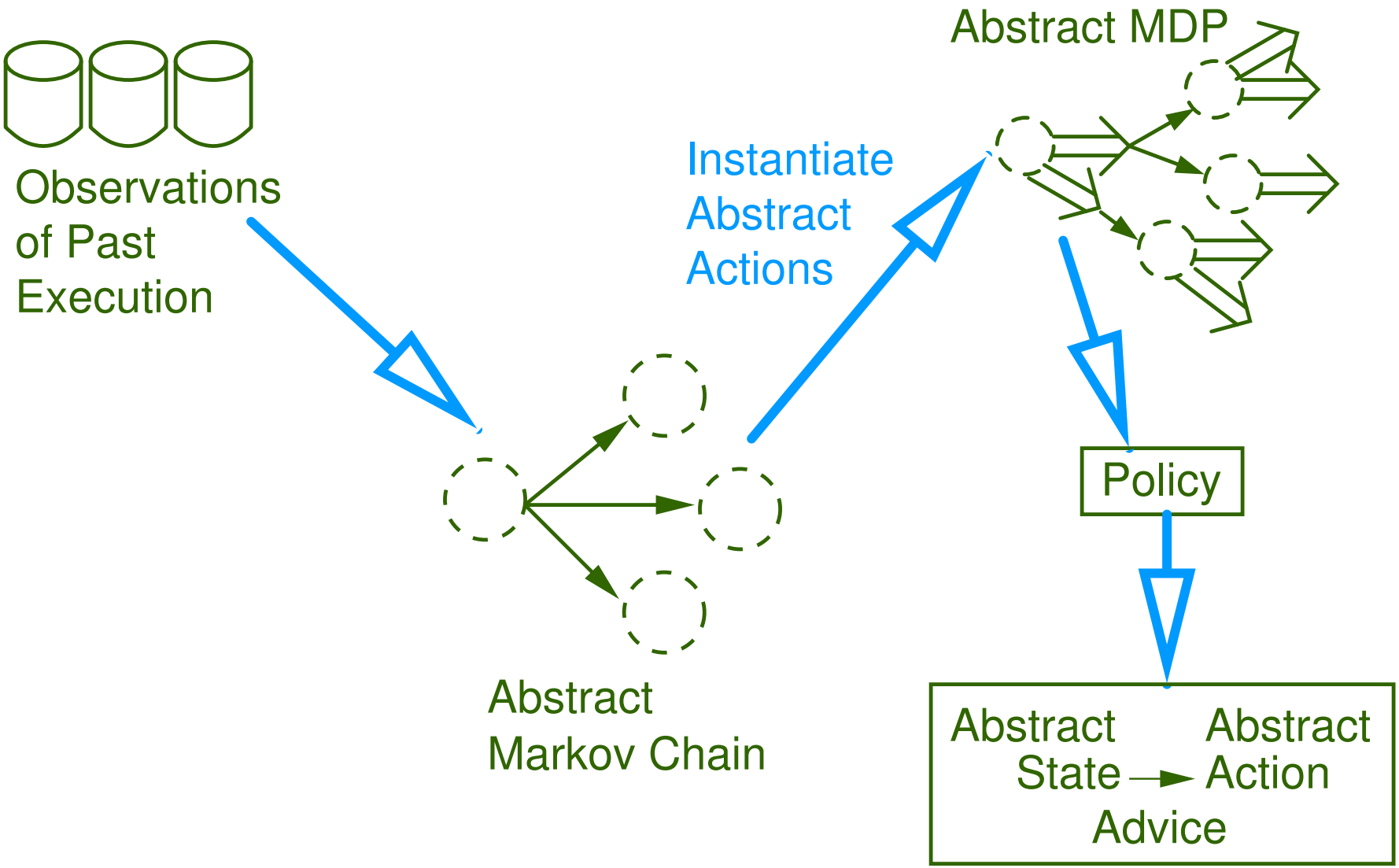
Markov Chain to MDP: Example



Markov Chain to MDP: Example



Observations, MC, MDP, Policy, Advice



Adding Rewards

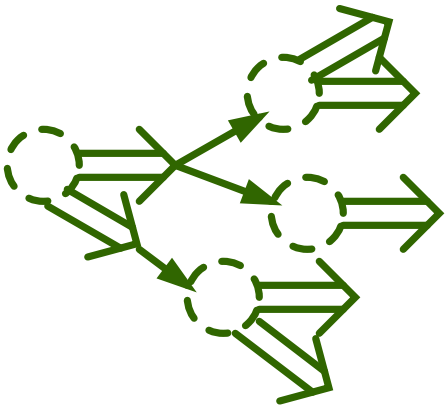
- We have learned an abstract transition model
  - MDP is currently reward-less

Adding Rewards

- We have learned an abstract transition model
  - MDP is currently reward-less
- Model can not be solved for an action policy until rewards are added
- The same transition model can be used for many different reward signals

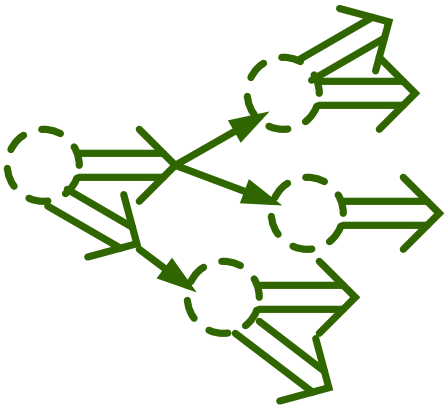
MDP to Advice

MDP



MDP to Advice

MDP

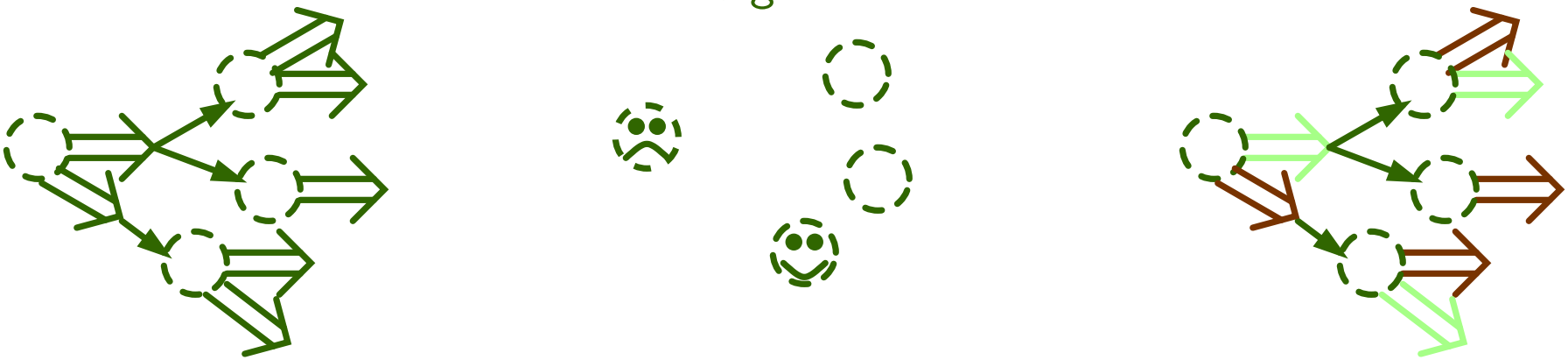


+

Reward
Signal

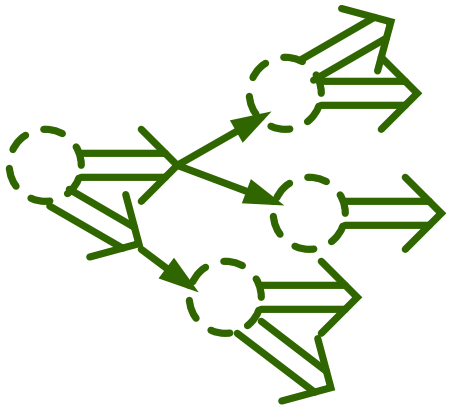
MDP to Advice

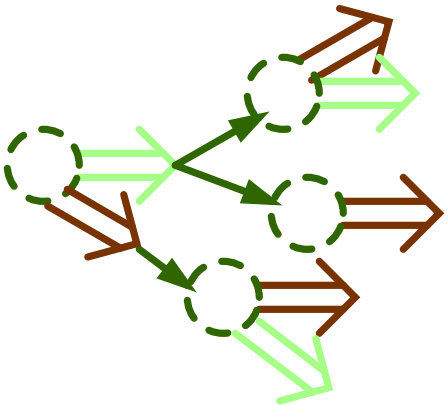
MDP + Reward Signal = Policy

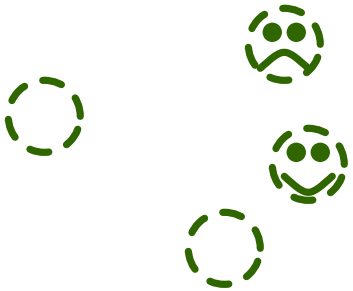


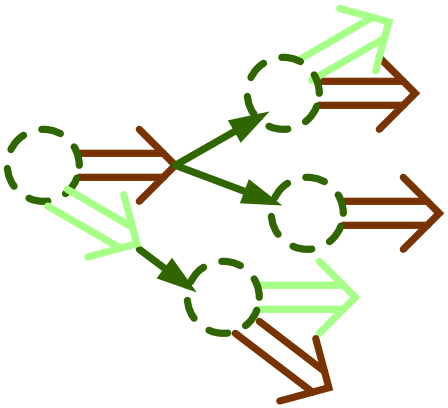
MDP to Advice

MDP + Reward Signal = Policy









Formalism

$\langle \bar{S}, T_{MC} \rangle$

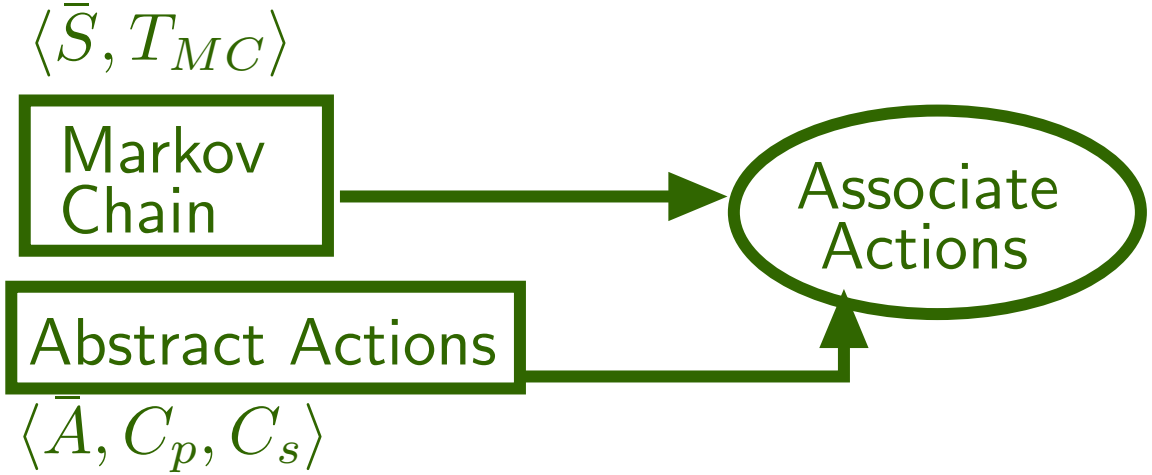
Markov Chain

$\bar{\mathcal{S}}$ Set of abstract states

$\bar{\mathcal{A}}$ Set of abstract actions

C_p, C_s Primary, Secondary transition descriptions

Formalism

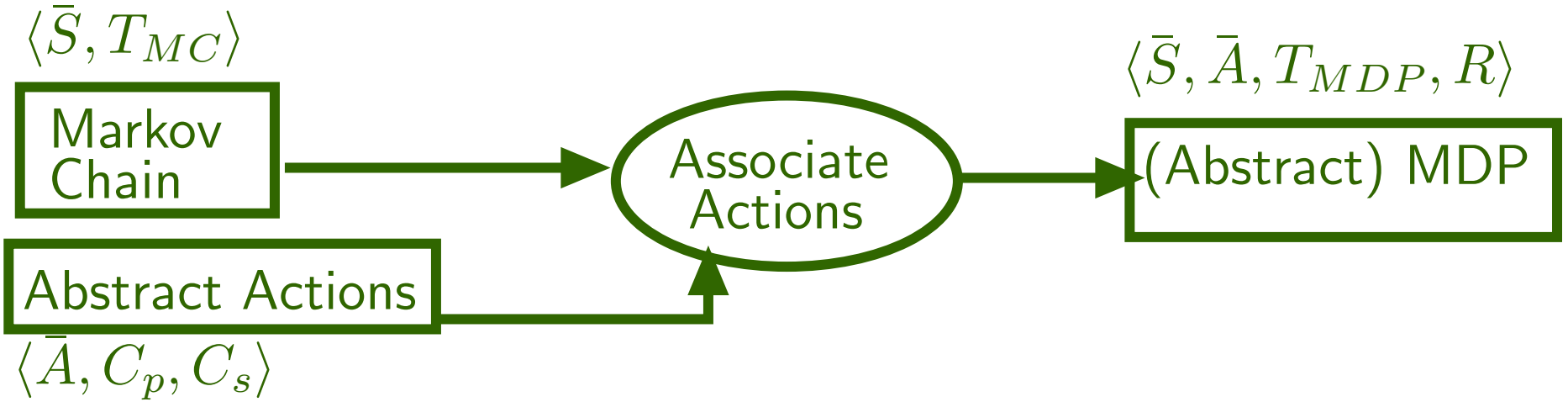


$\bar{\mathcal{S}}$ Set of abstract states

$\bar{\mathcal{A}}$ Set of abstract actions

C_p, C_s Primary, Secondary transition descriptions

Formalism

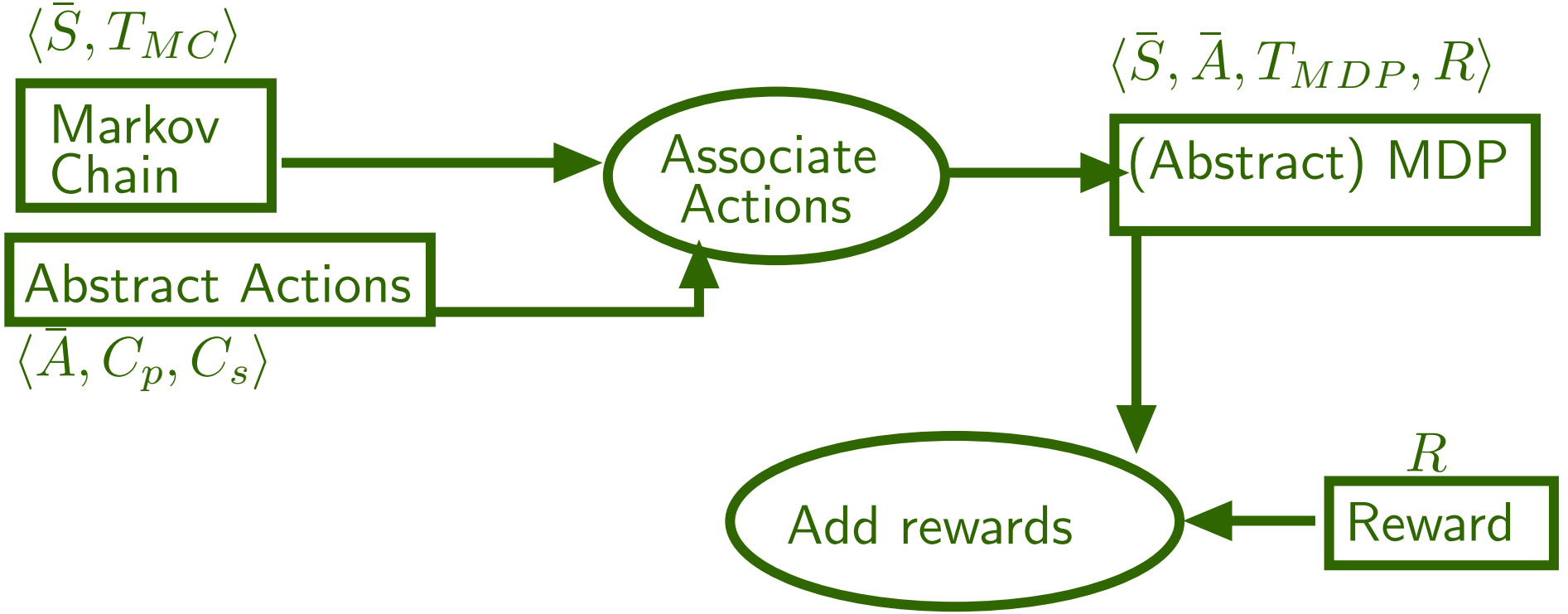


$\bar{\mathcal{S}}$ Set of abstract states

$\bar{\mathcal{A}}$ Set of abstract actions

C_p, C_s Primary, Secondary transition descriptions

Formalism

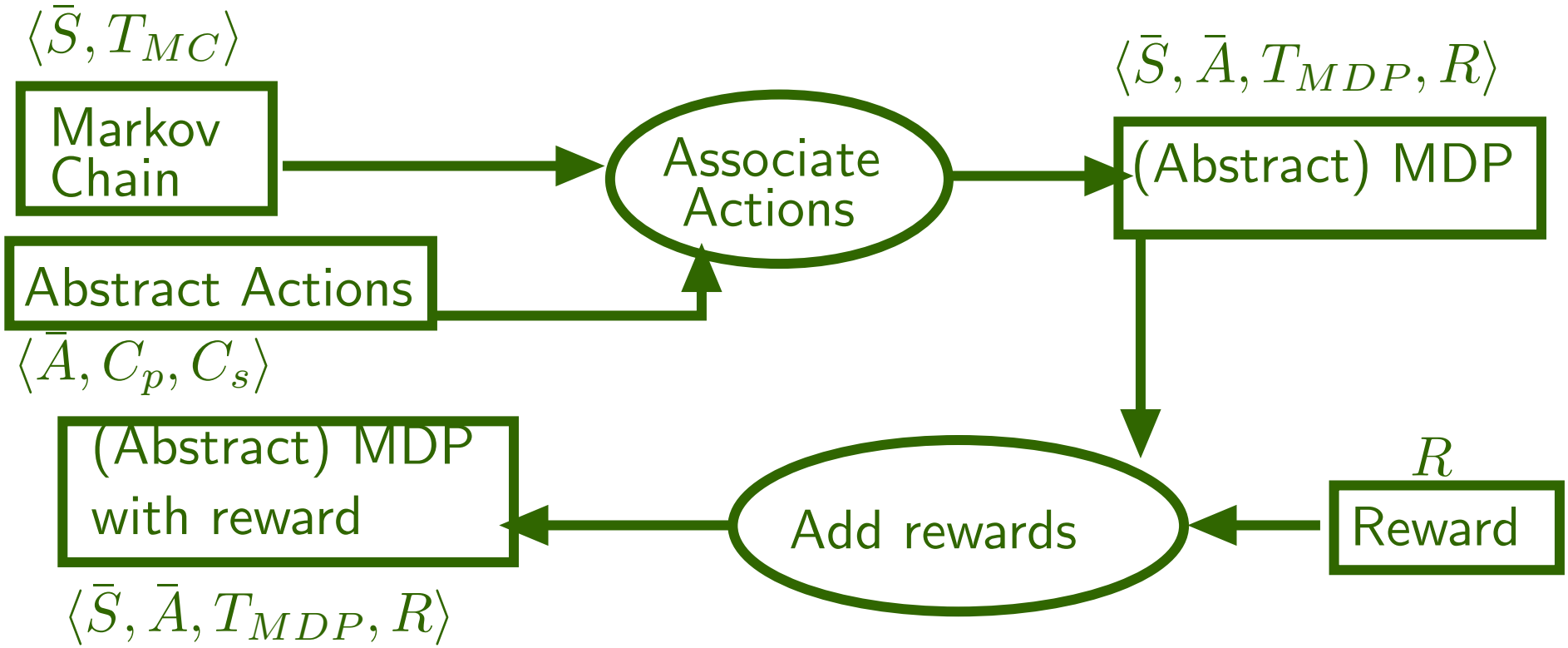


$\bar{S}$ Set of abstract states

$\bar{A}$ Set of abstract actions

C_p, C_s Primary, Secondary transition descriptions

Formalism



$\bar{S}$ Set of abstract states

$\bar{A}$ Set of abstract actions

C_p, C_s Primary, Secondary transition descriptions

Empirical: Flawed Opponent

Can our learning algorithm exploit an opponent's strategy?

Empirical: Flawed Opponent

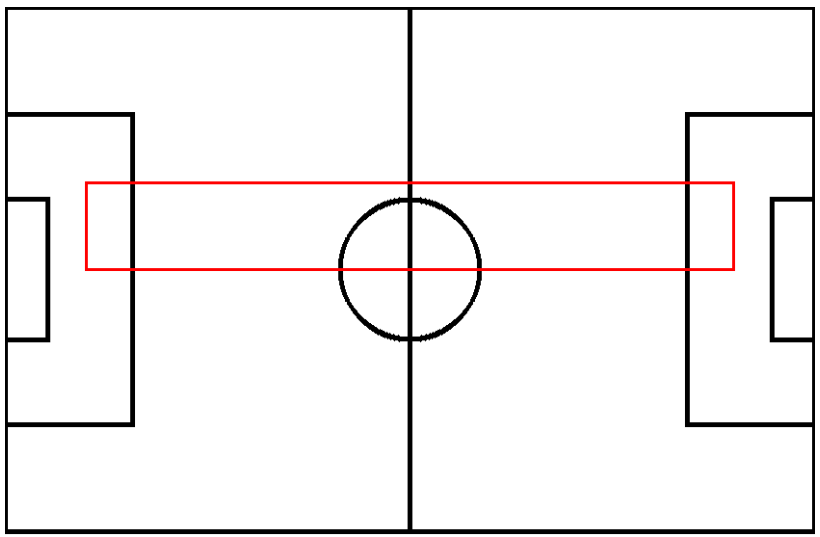
Can our learning algorithm exploit an opponent's strategy?

- Test against a team with a flaw that we program
  - Known set of states and actions will have high value

Empirical: Flawed Opponent

Can our learning algorithm exploit an opponent's strategy?

- Test against a team with a flaw that we program
  - Known set of states and actions will have high value
- Opponent team (on right) will not go into corridor below



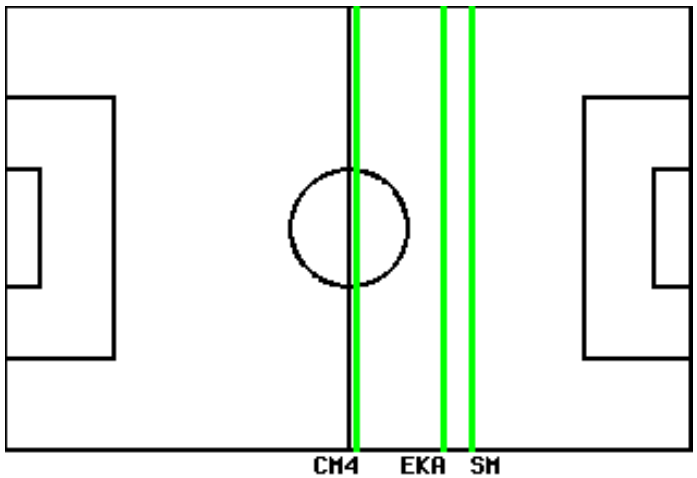
Empirical: Flawed Opponent Results

	Training		
Team	Score Difference	Mean Ball X	% Attacking
SM			
EKA			
CM4			

Empirical: Flawed Opponent Results

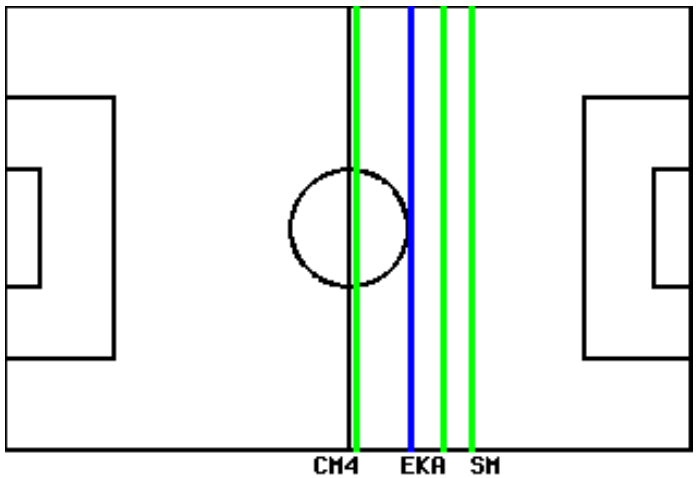
Training

Team	Score Difference	Mean Ball X	% Attacking
SM	12.2 $[11.3, 13.2]$	19.0 $[18.93, 19.11]$	43%
EKA	7.3 $[6.5, 8.1]$	14.6 $[14.47, 14.65]$	35%
CM4	0.7 $[0.4, 1.0]$	1.1 $[1.04, 1.16]$	24%



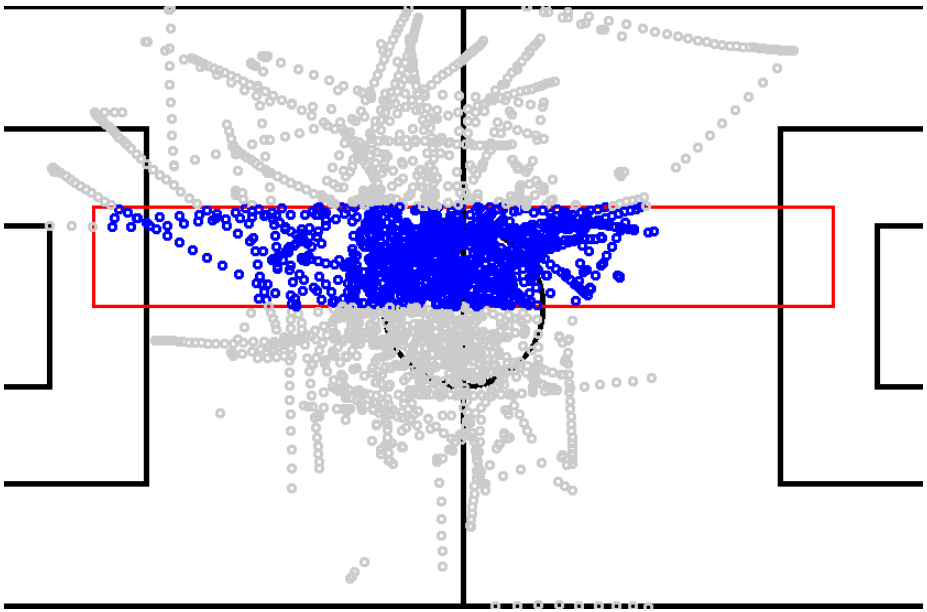
Empirical: Flawed Opponent Results

Team	Score Difference	Mean Ball X	% Attacking
Training			
SM	12.2 $[11.3, 13.2]$	19.0 $[18.93, 19.11]$	43%
EKA	7.3 $[6.5, 8.1]$	14.6 $[14.47, 14.65]$	35%
CM4	0.7 $[0.4, 1.0]$	1.1 $[1.04, 1.16]$	24%
Testing			
CM4	3.1 $[2.5, 3.7]$	9.5 $[9.46, 9.64]$	35%



Empirical: Flawed Opponent Results

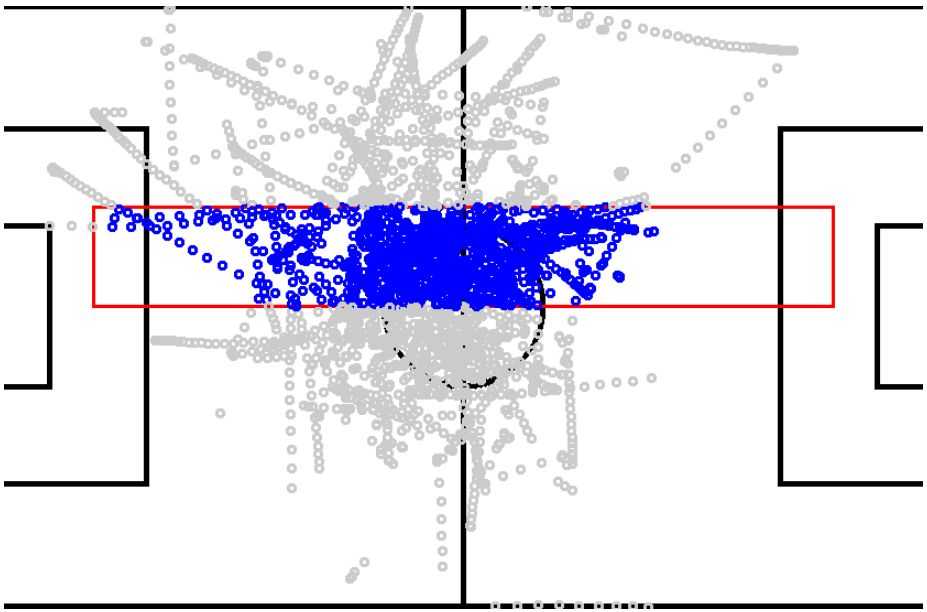
- Each dot represents a location of the ball when our team owned the ball



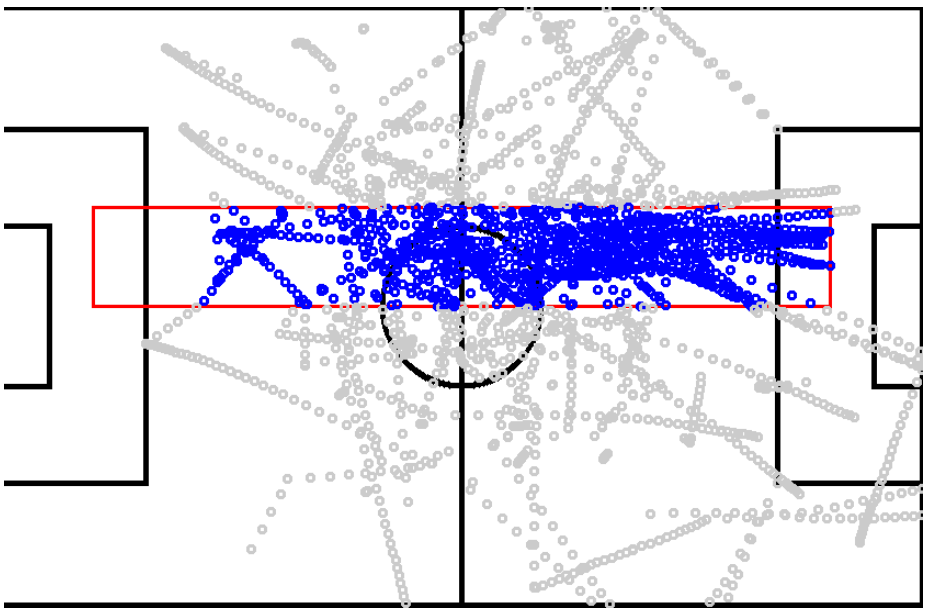
Training

Empirical: Flawed Opponent Results

- Each dot represents a location of the ball when our team owned the ball



Training



Testing

Soccer is Complicated!

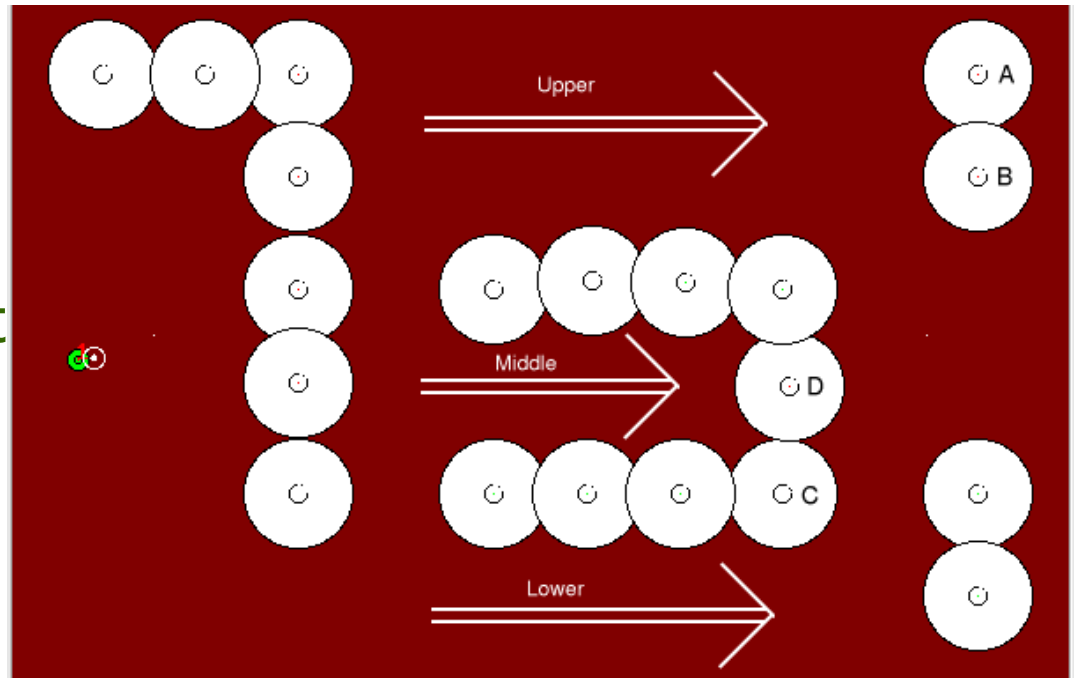
Soccer is Complicated!

- Team of advice receivers
- Team of opponents
- Infrequent, hard to achieve reward
  - Unclear evaluation metrics
- Unknown optimal policy

Introducing RCSSMaze

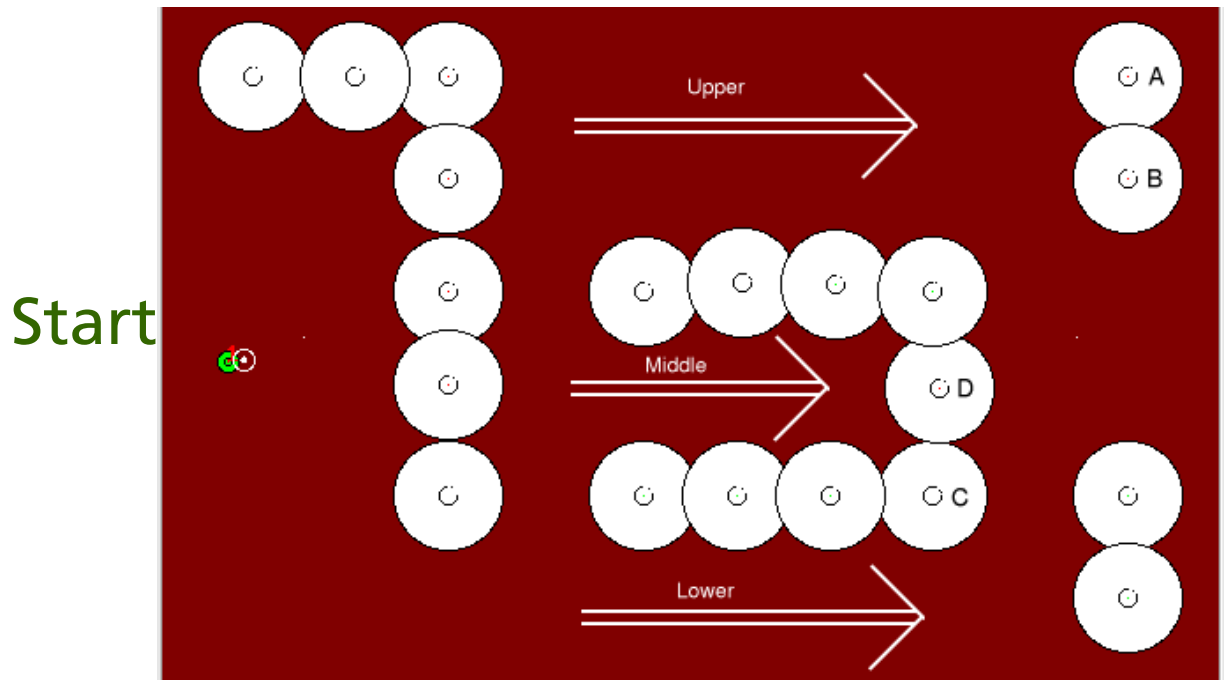
- Continuous state/action spaces, partial observability

Start



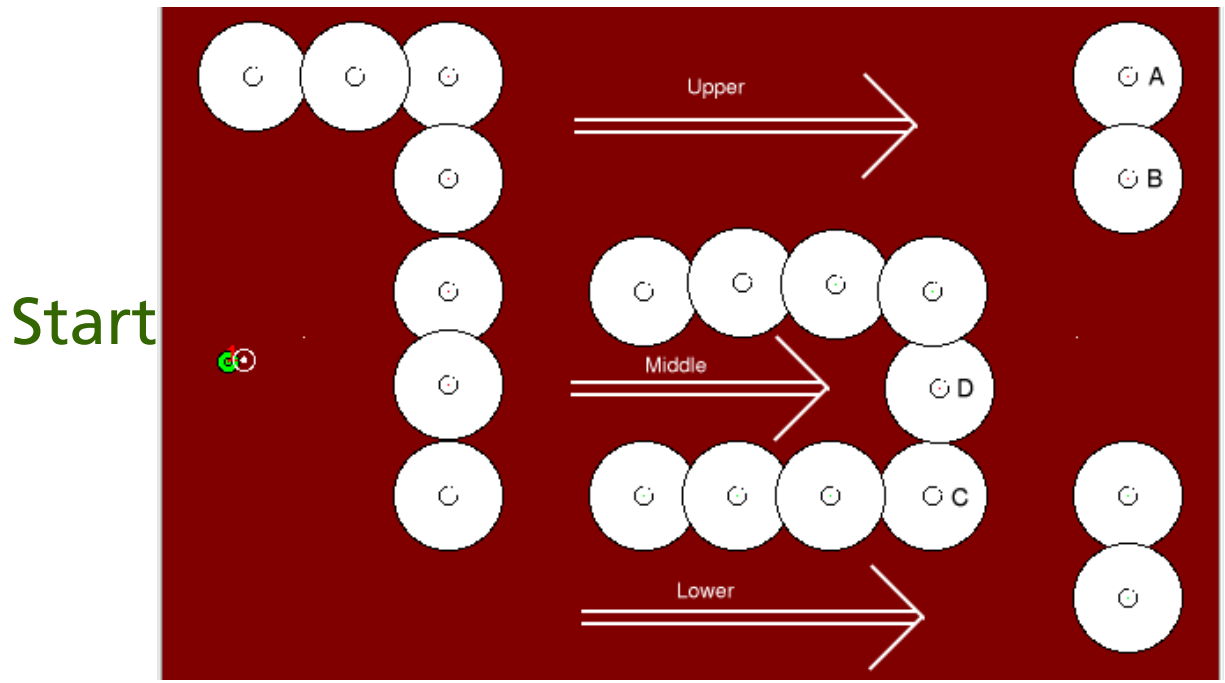
Introducing RCSSMaze

- Continuous state/action spaces, partial observability
- Single executing agent receiving advice
  - "Wall" agents execute fixed movement behaviors



Introducing RCSSMaze

- Continuous state/action spaces, partial observability
- Single executing agent receiving advice
  - "Wall" agents execute fixed movement behaviors
- We approximately know the optimal policy



RCSSMaze Training

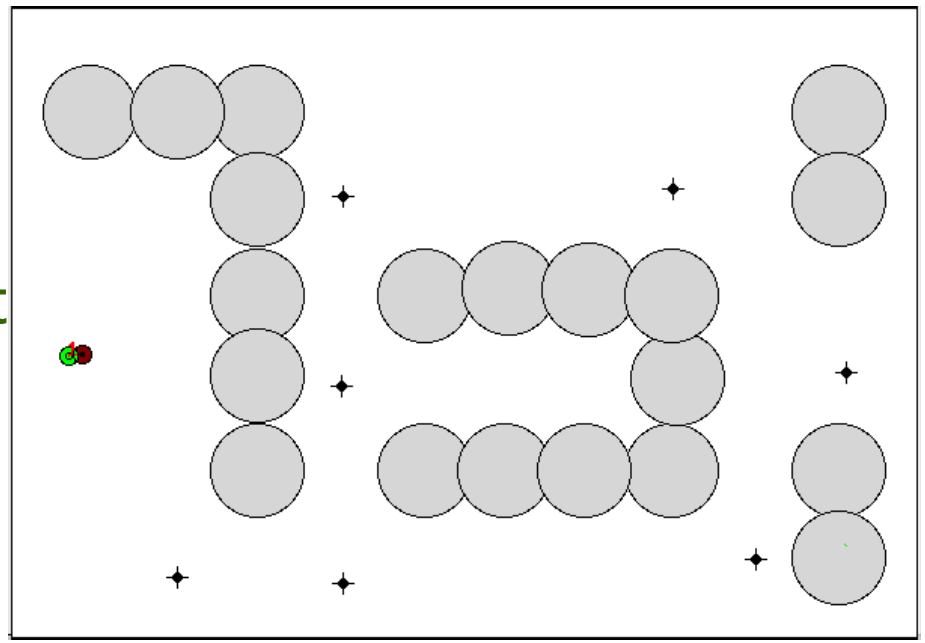
Can our algorithm learn a model for effective advice?

RCSSMaze Training

Can our algorithm learn a model for effective advice?

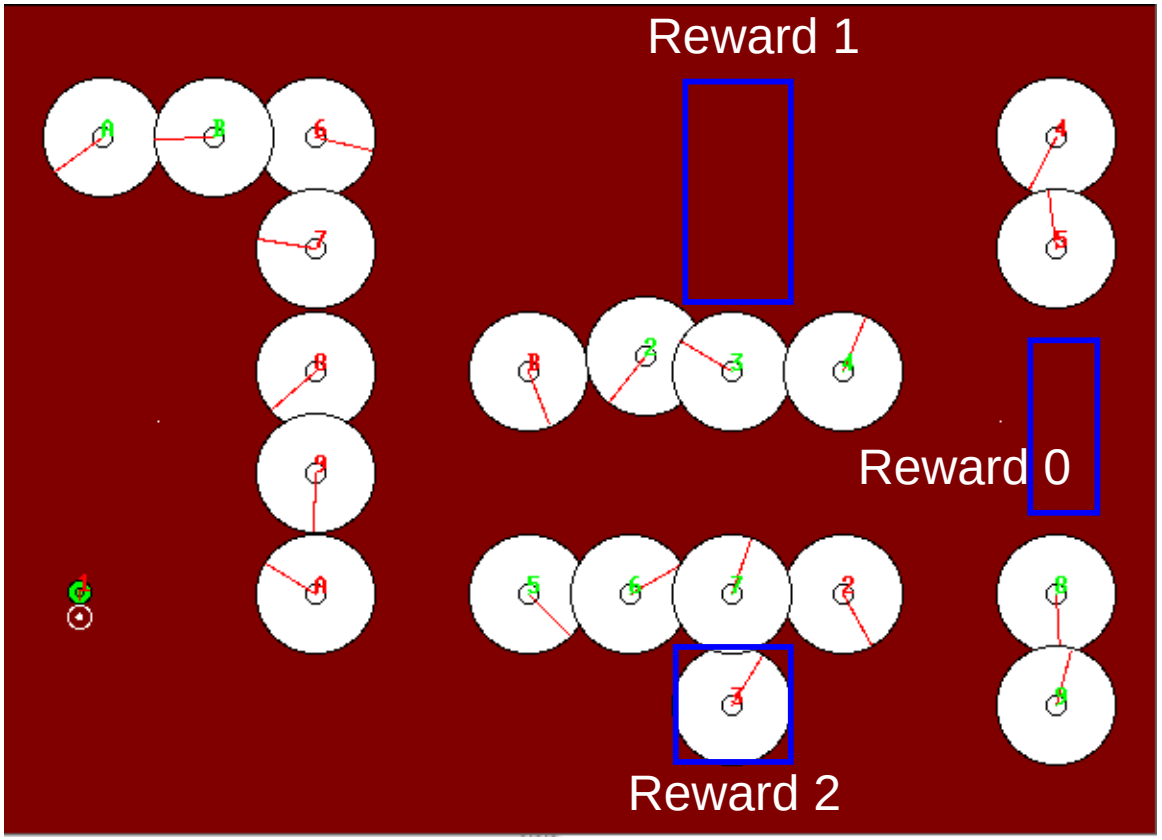
- Training data (240 minutes)
  - Agent randomly picks one of given points
  - Heads directly to point until reached or reset to start
  - 5% of time, heads in a random direction

Start



RCSSMaze Rewards

- We can put reward wherever we want



RCSSMaze Results

- A trial begins when the agent is at the start state
- A trial ends when
  - A positive reward is received
  - The agent is reset to the start state
- A successful trial is one that receives positive reward

RCSSMaze Results

- A trial begins when the agent is at the start state
- A trial ends when
  - A positive reward is received
  - The agent is reset to the start state
- A successful trial is one that receives positive reward

Reward	% in Training	% with MDP
0	$< 1\%$	64%
1	1%	60%
2	7%	93%

MDP Learning and Other Domains

- We used the MDP for advice, but environment models are useful in other contexts

MDP Learning and Other Domains

- We used the MDP for advice, but environment models are useful in other contexts

- Algorithm inputs
  - External observations (do **not** need to see inside agents' heads)
  - Abstract state space
  - Abstract action templates

MDP Learning and Other Domains

- We used the MDP for advice, but environment models are useful in other contexts
- Algorithm inputs
  - External observations (do **not** need to see inside agents' heads)
  - Abstract state space
  - Abstract action templates
- Apply any reward function

Summary and Previous and Future Work

Coaching and Previous Work

Intelligent Tutoring Systems

- Systems to instruct human students
- Generally used with complete and correct expert model
- Focused on humans

Coaching and Previous Work

Intelligent Tutoring Systems

- Systems to instruct human students
- Generally used with complete and correct expert model
- Focused on humans

Agents Taking Advice

- Lots of Reinforcement Learning [e.g. Maclin and Shavlik, 1996]
- How to operationalize advice? [e.g. Mostow, 1981]
- Use some similar techniques to incorporate advice, but real concern is **giving** advice

Coaching and Previous Work

Abstract/Factored Markov Decision Processes

- Efficient reasoning by learning/using abstractions [e.g. Dearden and Boutilier, 1997, Uther and Veloso, 2002]
- Factored representations [Dean and Kanazawa, 1989] and their applications [e.g. Guestrin et al., 2001]

Coaching and Previous Work

Abstract/Factored Markov Decision Processes

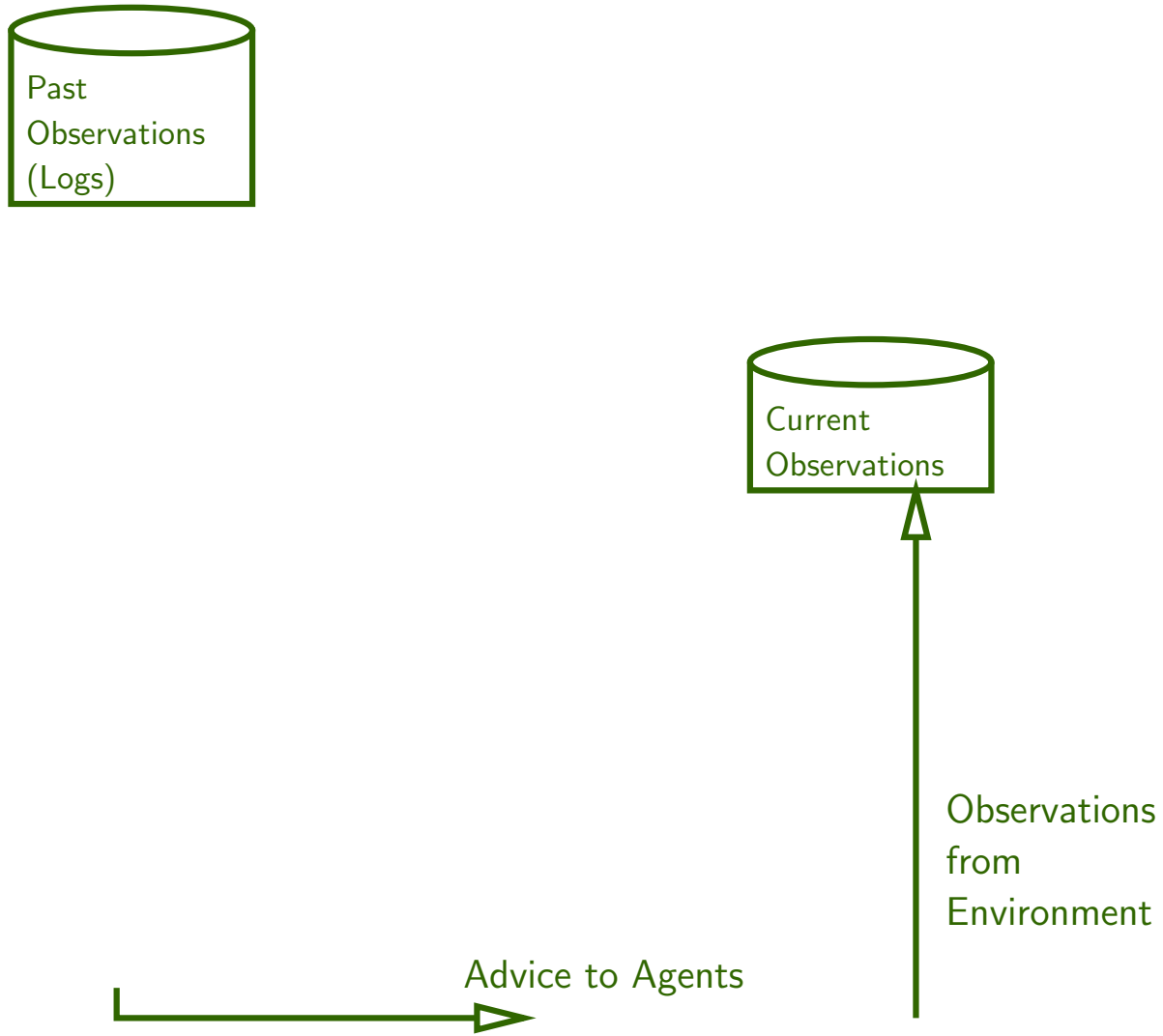
- Efficient reasoning by learning/using abstractions [e.g. Dearden and Boutilier, 1997, Uther and Veloso, 2002]
- Factored representations [Dean and Kanazawa, 1989] and their applications [e.g. Guestrin et al., 2001]

Coaching in Robot Soccer

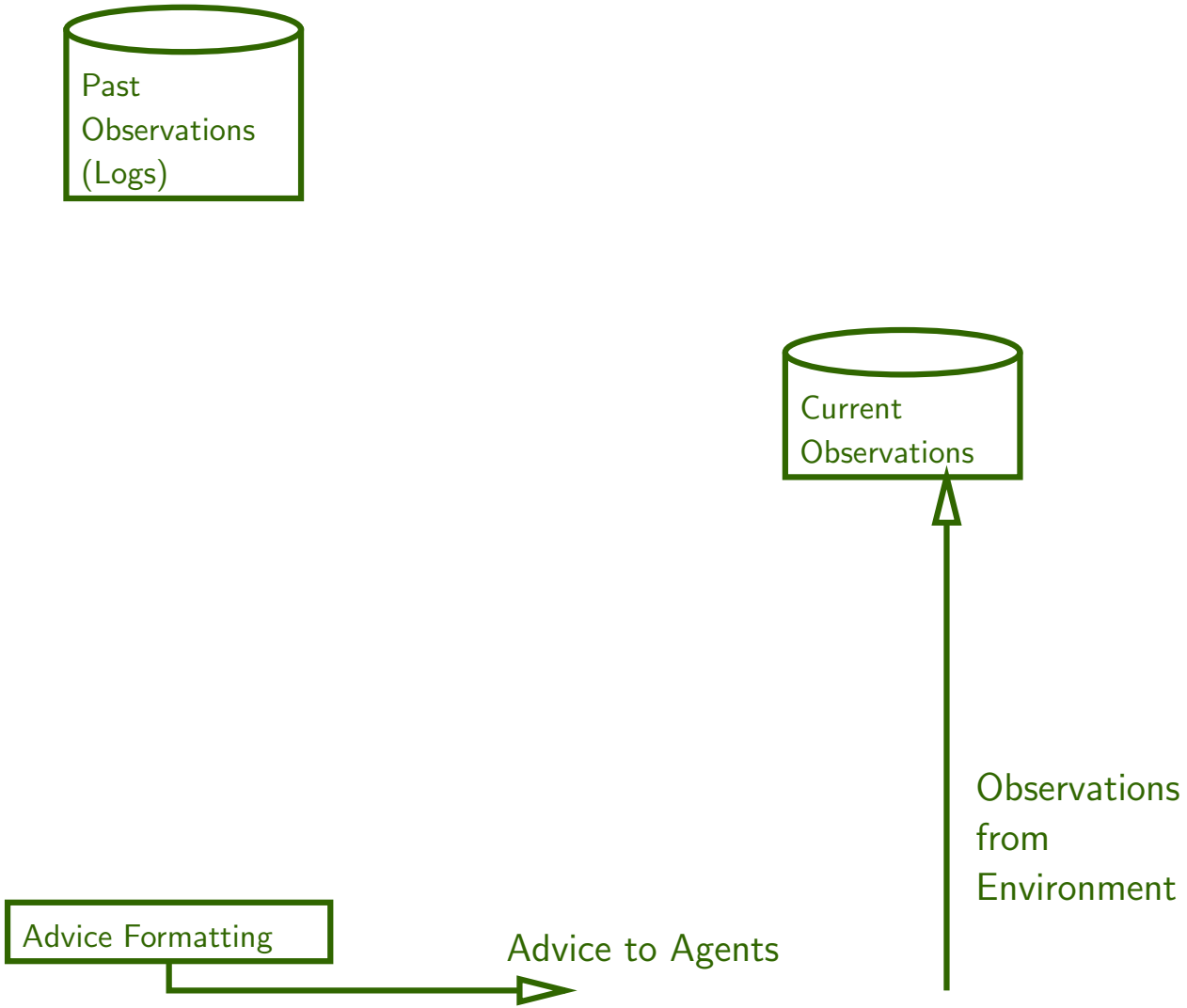
- This thesis grew with and helped define this field
- Early coaching work dealt with formations [Takahashi, 2000]
- ISAAC [Raines et al., 2000]
- Opponent modeling [Steffens, 2002, Kuhlmann et al., 2004]

Big Picture Summary

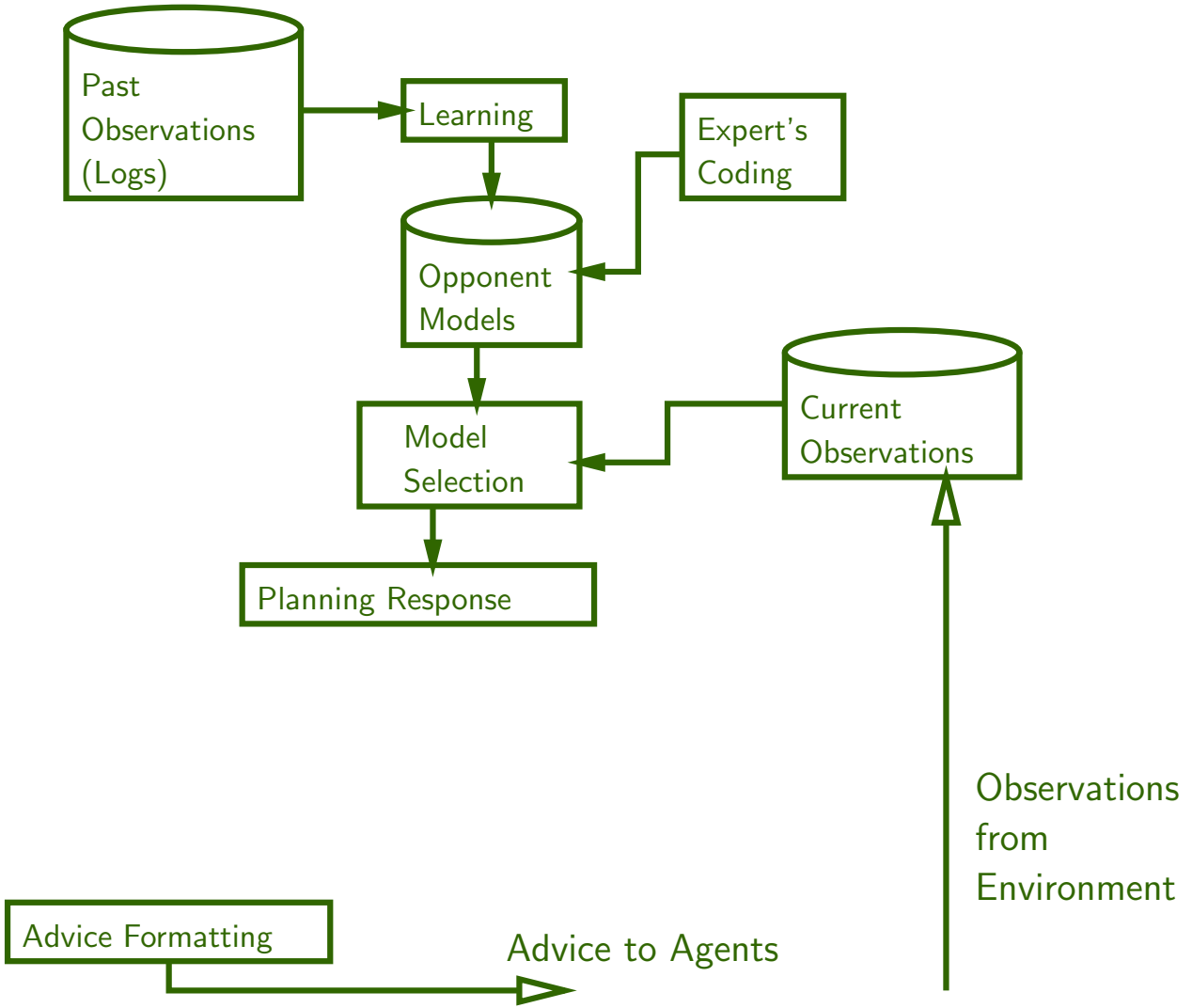
Big Picture Summary



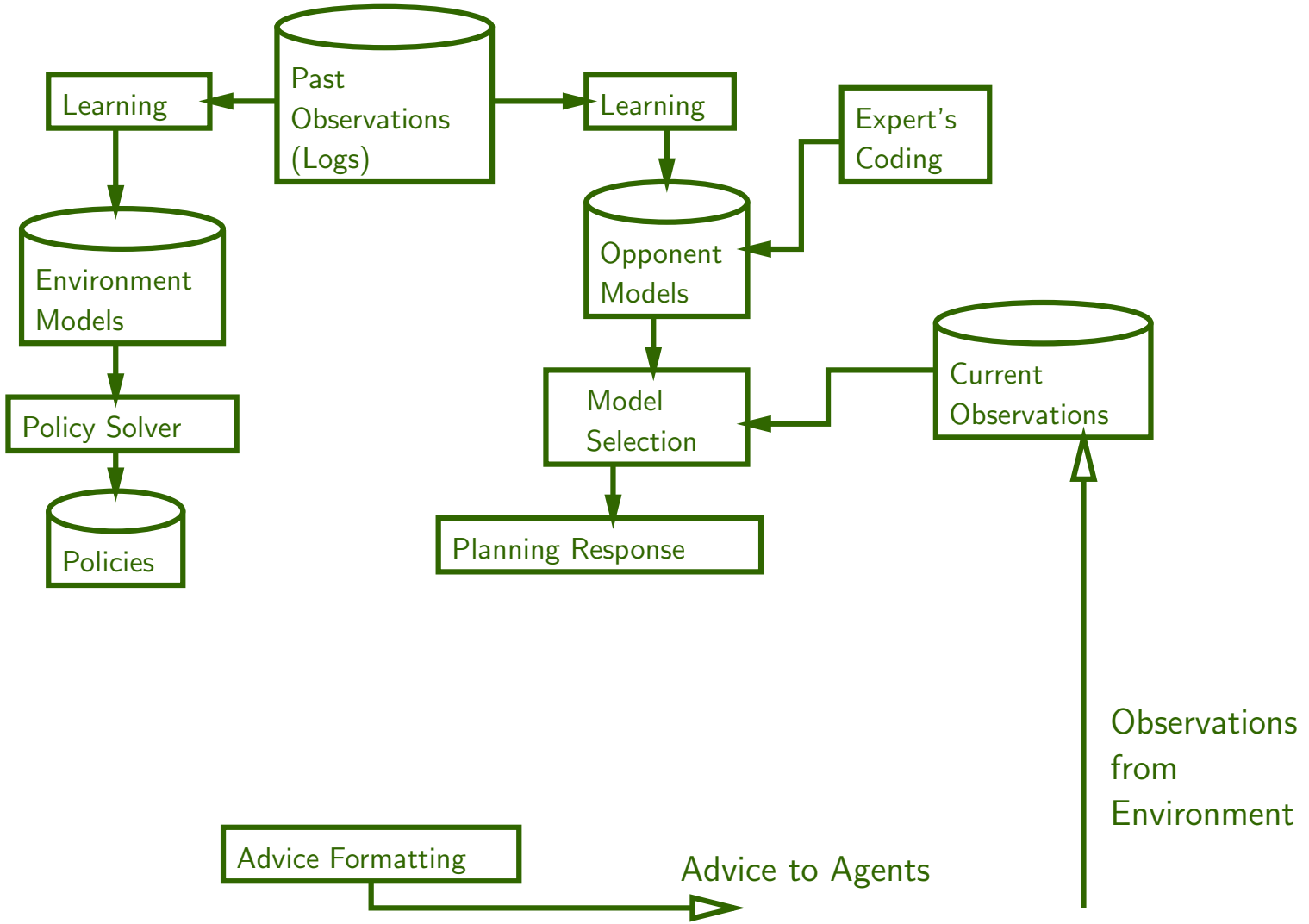
Big Picture Summary



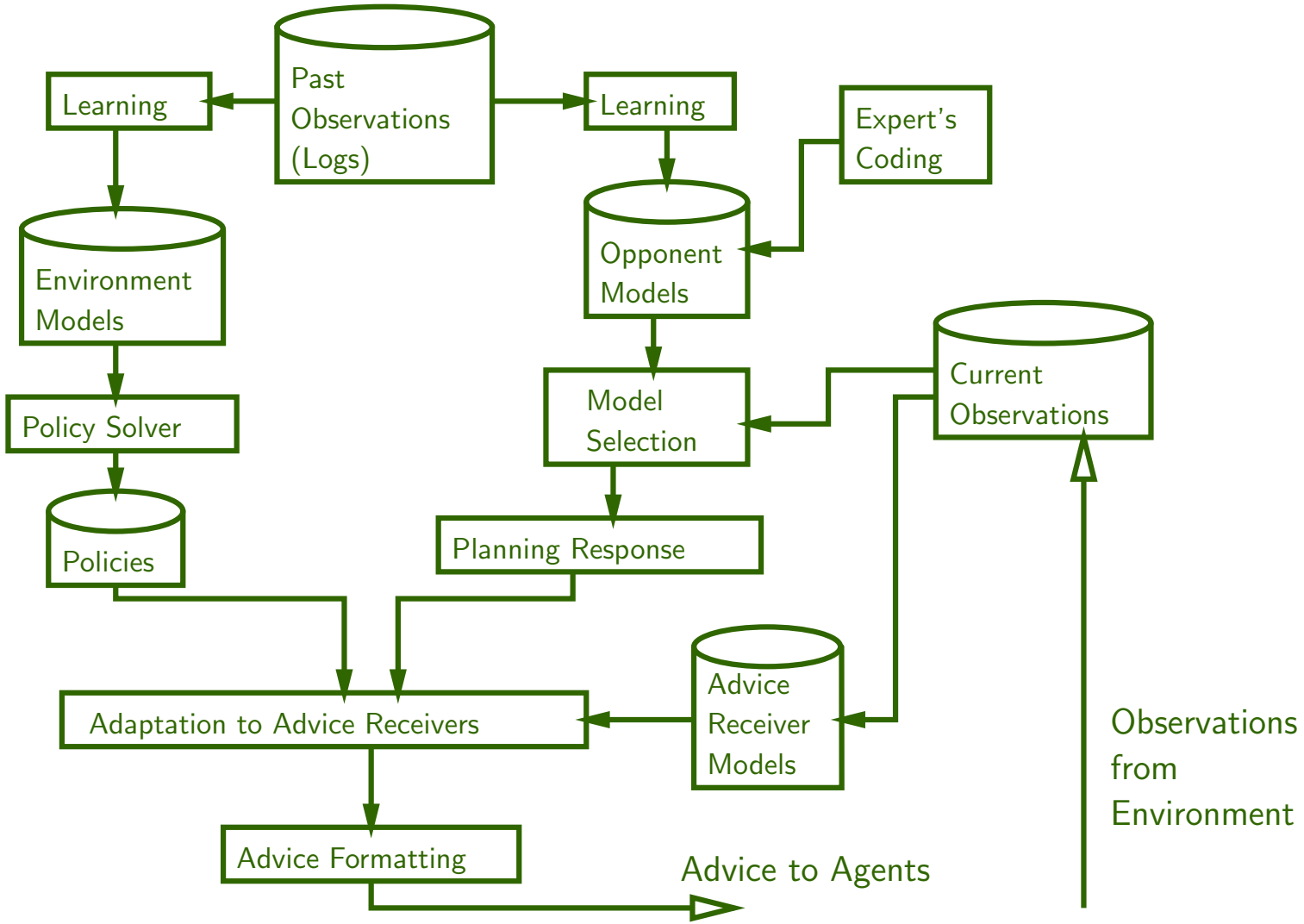
Big Picture Summary



Big Picture Summary



Big Picture Summary



Contributions

- Several opponent model representations, with learning and advice generation algorithms (in robot soccer)
- Algorithms for learning an abstract MDP from observations, given state abstraction, and abstract action templates
- Study of adapting advice in a predator-prey environment considering limitation and communication bandwidth
- Multi-Agent Simple Temporal Networks: novel multi-agent plan representation and accompanying execution algorithm
- Largest empirical study of coaching in simulated robot soccer (5000 games/2500 hours)

Future Work: Abstract MDP Learning

- Recursive Learning and Verification of Abstract Markov Decision Processes
- Learning Hierarchical Semi-Markov Decision Processes from External Observation
- Refining State Abstractions for Markov Decision Process Learning

Future Work: Adapting to Advice Receivers

- Learning About Agents While Giving Advice
- Talking Back: How Advice Receivers Can Help Their Coaches
- What I See and What I Don't: What a Coach Needs to Know About Partial Observability

Questions?

Why is the Coach a Separate Agent?

- Some of the reasoning described could be done by a single executing agent
- Advice language provides abstraction to work across agents
- Agent systems **will** be more distributed

Why Coaching?

Disclaimer: This isn't a philosophy talk

Coach/agent separation is a forced distribution

- Why would/should one make their agent system like this?
- Agent systems **will** be more distributed — how will agents interact?
- Knowledge transfer will not always be easy

Coaching Problem Properties

- Team goals
- External, observing coach
- Advice, not control
- Access to past behavior logs
- Advice at execution, not training

Coaching Problem Dimensions

- Online vs. offline learning
- One-time vs occasional vs. continual advice
- Advice as actions vs. macro-actions vs. plans

Coaching General Lessons

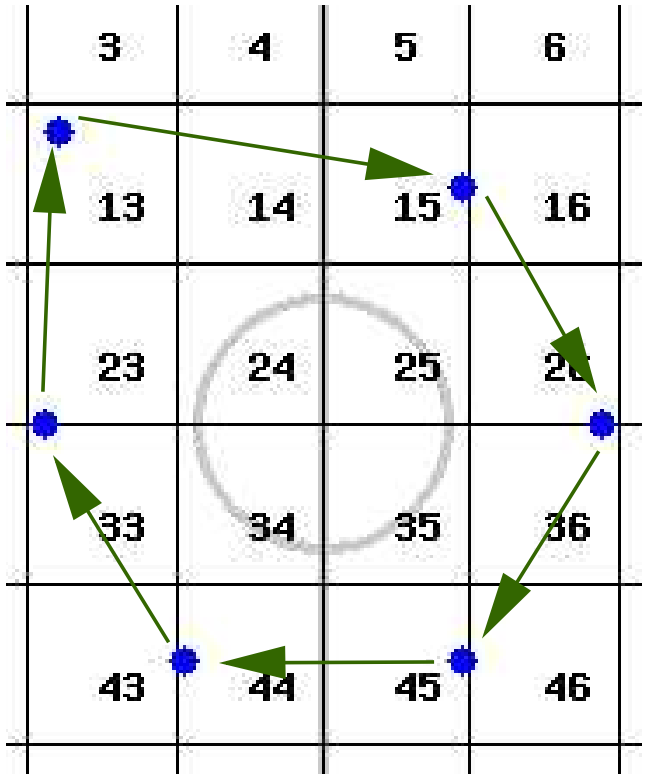
- The coach and advice receivers are a tightly coupled system
- Coach learning will require iteration to achieve the best performance
- A tradeoff exists in how much of the state space to cover with advice versus how good the advice is
- Different observability by the coach and agents can be ignored somewhat, but will need to be considered at times
- Analyzing the past behavior of an agent is most useful only if the future will look similar to the past

Empirical: Circle Passing

- By using a domain smaller than the whole soccer game, can better isolate effects
- Setup
  - Give the players a fixed action strategy
  - Because of noise, coach will see other possible action results
- Coach learns a model, then gives advice
- Different rewards lead to different agent behaviors

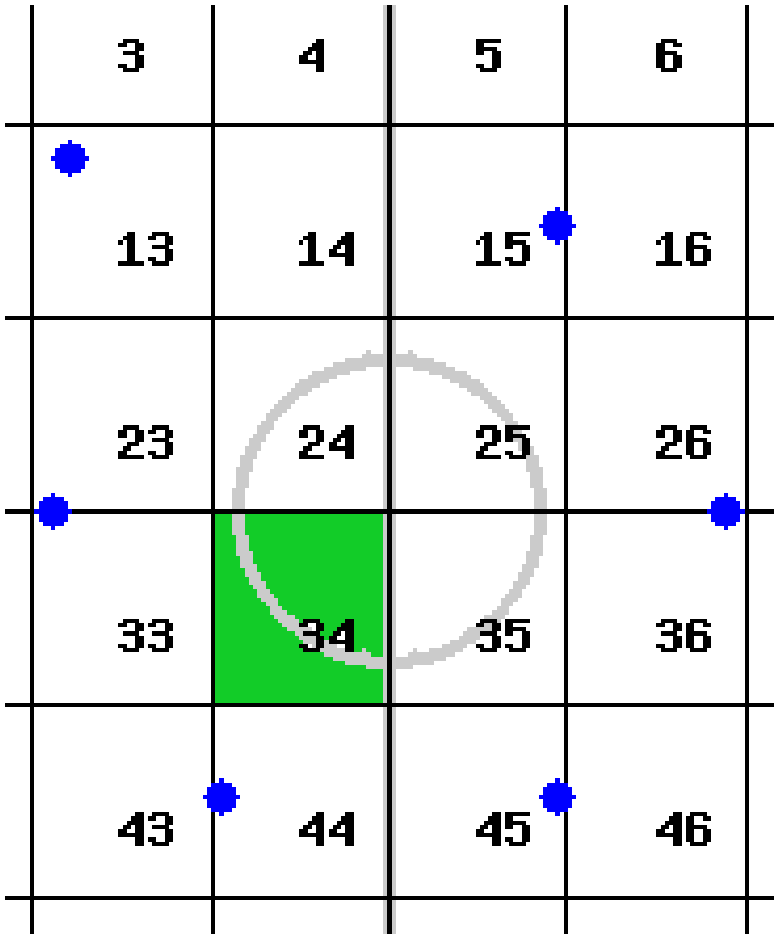
Circle Passing: Setup

- Six players trying to pass in a circle
- Not all passes are successful
- Some kicks result in passes to other players or a dribble



Circle Passing: Reward

- Can apply any reward function
- We'll describe one (more in the thesis)
- In the middle (miskicks from several players go here)



Circle Passing: Results

- We consider a trial a success if:
  - From a random starting position
  - Reward is received within 200 cycles (20 seconds)

Success % During Training	40%
Success % With Advice	88%

RCSSMaze: Recursive Learning

Training Data	# Rew. (Training) R_0	R_1	R_2	% Success (Testing) R_0	R_1	R_2
Original	11	115	1055	64%	60%	93%
From R_0	676	0	0	82%	n/a	n/a
From R_1	1	2909	0	0%	67%	n/a
From R_2	0	0	9088	n/a	n/a	78%

References

T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 76(1–2):3–74, 1989.

Richard Dearden and Craig Boutilier. Abstraction and approximate decision theoretic planning. *Artificial Intelligence*, 89(1): 219–283, 1997.

Diana Gordon and Devika Dubramanian. A multi-strategy learning scheme for knowledge assimilation in embedded agents. *Informatica*, 17, 1993.

Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored mdps. In *Advances in Neural Information Processing Systems 14*, 2001.

Gregory Kuhlmann, Peter Stone, and Justin Lallinger. The champion UT Austin Villa 2003 simulator online coach team. In Daniel Polani, Brett Browning, Andrea Bonarini, and Kazuo Yoshida, editors, *RoboCup-2003: Robot Soccer World Cup VII*. Springer Verlag, Berlin, 2004. To appear.

Richard Maclin and Jude W. Shavlik. Creating advice-taking reinforcement learners. *Machine Learning*, 22:251–282, 1996.

Jack Mostow. *Mechanical Transformation of Task Heuristics into Operational Procedures*. PhD thesis, Carnegie Mellon University, 1981.

Taylor Raines, Milind Tambe, and Stacy Marsella. Automated assistant to aid humans in understanding team behaviors. In *Proceedings of the Fourth International Conference on Autonomous Agents (Agents-2000)*, 2000.

Timo Steffens. Feature-based declarative opponent-modelling in multi-agent systems. Master's thesis, Institute of Cognitive Science Osnabrück, 2002. URL citeseer.nj.nec.com/steffens02featurebased.html.

Tomoichi Takahashi. Kasugabito III. In Veloso, Pagello, and Kitano, editors, *RoboCup-99: Robot Soccer World Cup III*, number 1856 in Lecture Notes in Artificial Intelligence, pages 592–595. Springer-Verlag, Berlin, 2000.

William Uther and Manuela Veloso. TTree: Tree-based state generalization with temporally abstract actions. In *Proceedings of SARA-2002*, Edmonton, Canada, August 2002.