Multiagent Reactive Plan Application Learning in Dynamic Environments

Hüseyin Sevay
Department of Electrical Engineering and Computer Science

University of Kansas

`hsevay@eecs.ku.edu`

May 3, 2004

Overview

Introduction

- Theme: How to solve problems in realistic multiagent environments?
- Problem
  - Complex, dynamic, uncertain environments have prohibitively continuous/large state and action spaces. Search spaces grow exponentially with the number of agents
  - Goals require multiple agents to accomplish them
  - Agents are autonomous and can only observe the world from their local perspectives and do not communicate
  - Sensor and actuator noise

Introduction (cont'd)

- Solution Requirements
  - Agents need to collaborate among themselves
  - Agents must coordinate their actions
- Challenges
  - How to reduce large search spaces
  - How to enable agents to collaborate and coordinate to exhibit durative behavior
  - How to handle noise and uncertainty

Introduction (cont’d)

- *Mu*ltiagent *R*eactive plan *A*pplication *L*earning (MuRAL)
  - proposed and developed a *learning-by-doing* solution methodology that
    - ∗ uses high-level plans to focus search from the perspective of each agent
    - ∗ each agent learns independently
    - ∗ facilitates goal-directed collaborative behavior
    - ∗ uses case-based reasoning and learning to handle noise
    - ∗ incorporates a naive form of reinforcement learning to handle uncertainty

Introduction (cont'd)

- We implemented MuRAL agents that work in the RoboCup soccer simulator
- Experimentally we show that learning improves agent performance
  - Learning becomes more critical as plans get more complex

Terminology

- **agent:** an entity (hardware or software) that has its own decision and action mechanisms
- **plan:** a high-level description of *what* needs to be done by a team of agents to accomplish a shared goal
- **dynamic:** continually and frequently changing
- **reactive:** responsive to dynamic changes
- **complex:** with very large state and action search spaces
- **uncertain:** difficult to predict
- **role:** a set of responsibilities in a given plan step

Motivation for Learning

- Complex, dynamic, uncertain environments require adaptability
  - balance *reaction* and *deliberation*
  - accomplish goals despite adversities
- Interdependencies among agent actions are context-dependent
- Real-world multiagent problem solving requires methodologies that account for durative actions in uncertain environments
  - a strategy may require multiple agents and may last multiple steps
  - Example: two robots pushing a box from point A to point B

Background

- Traditional planning (deliberative systems)
  - Search to transform the *start state* into *goal state*
  - Only source of change is the planner
- Reactive/Behavior-based systems
  - Map situations to actions
- Procedural reasoning
  - Preprogrammed (complex) behavior
- Hybrid systems
  - Combine advantages of reactive systems and deliberative planning

Related Work

- Reinforcement Learning
  - Learn a policy/strategy over infinitely many trials
  - Only a single policy is learned
  - Convergence of learning is difficult

Assumptions

- High-level plans can be written for a given multiagent environment
- Goals can be decomposed into several coordinated steps for multiple roles

An Example Soccer Plan

```
(plan Singlepass
   (rotation-limit 120 15)
   (step 1
      (precondition
         (timeout 15)
         (role A 10
            (has-ball A)
            (in-rectangle-rel B 12.5 2.5 12.5 -2.5 17.5 -2.5 17.5 2.5))
         (role B -1
            (not has-ball B)
            (in-rectangle-rel A 17.5 -2.5 17.5 2.5 12.5 2.5 12.5 -2.5)) )
      (postcondition
         (timeout 40)
         (role A -1 (has-ball B))
         (role B -1 (has-ball B) (ready-to-receive-pass B)))
      (application-knowledge
         (case Singlepass A 10 . . .
            (action-sequence (pass-ball A B)) ) )
```

Approach

- Thesis question: *How can we enable a group of goal-directed autonomous agents with shared goals to behave collaboratively and coherently in complex, highly dynamic and uncertain domains?*
- Our system: *Mu*ltiagent *R*eactive plan *A*pplication *L*earning (MuRAL)
  - A *learning-by-doing* solution methodology for enabling agents to learn to apply high-level plans to dynamic, complex, and uncertain environments

Approach

- Instead of learning of strategies as in RL, learning of how to fulfill *roles* in high-level *plans* from each agent's local perspective using a knowledge-based methodology
- Start with high-level *skeletal* plans
- Two phases of operation to acquire and refine knowledge that implements the plans (*learning-by-doing*)
  - *application knowledge*

Approach (cont'd)

- Phases of operation
  - *Training*: each agent acquires knowledge about how to solve specific instantiations of the high-level problem in a plan for its own role (*top-down*)
    - case-based learning
  - *Application*: each agent refines its training knowledge to be able to select more effective plan implementations based on its experiences (*bottom-up*)
    - naive reinforcement learning

Methodology

- A *skeletal* plan contains a set of preconditions and postconditions for each plan step
  - describes only the conditions *internal* to a collaborative group
  - missing from a skeletal plan is the application knowledge for implementing the plan in specific scenarios
- Each role has *application knowledge* for each plan step stored in cases. A *case*
  - describes the scenario in terms of conditions *external* to the collaborative group
  - contains an action sequence to implement a given plan step
  - records the success rate of its application

Methodology: Training

```
for a skeletal plan, P
   for an agent, A, that takes on role r in P
      for each step, s, of P
         - A dynamically builds a search problem using
           its role description in s
         - A does search to implement r in s
           in terms of high-level actions
         - A executes the search result
         - if successful, A creates and stores a case
           that includes a description of the external
           environment and the search result
```

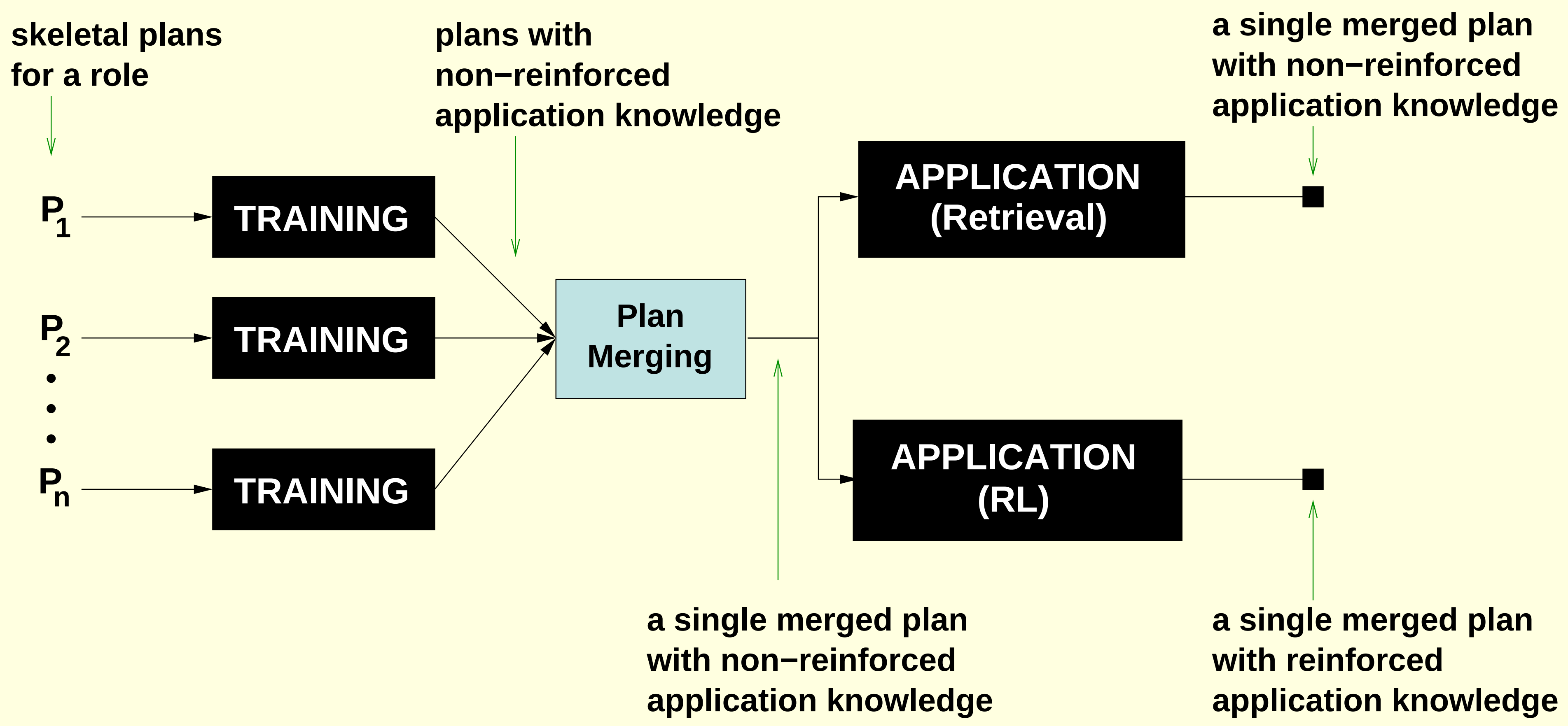
Methodology: Plan Merging

- For each successful training trial, each agent stores its application knowledge locally
- We merge all pieces of knowledge from successful training trials into a single plan for each *role* (postprocessing)
  - ignore duplicate solutions

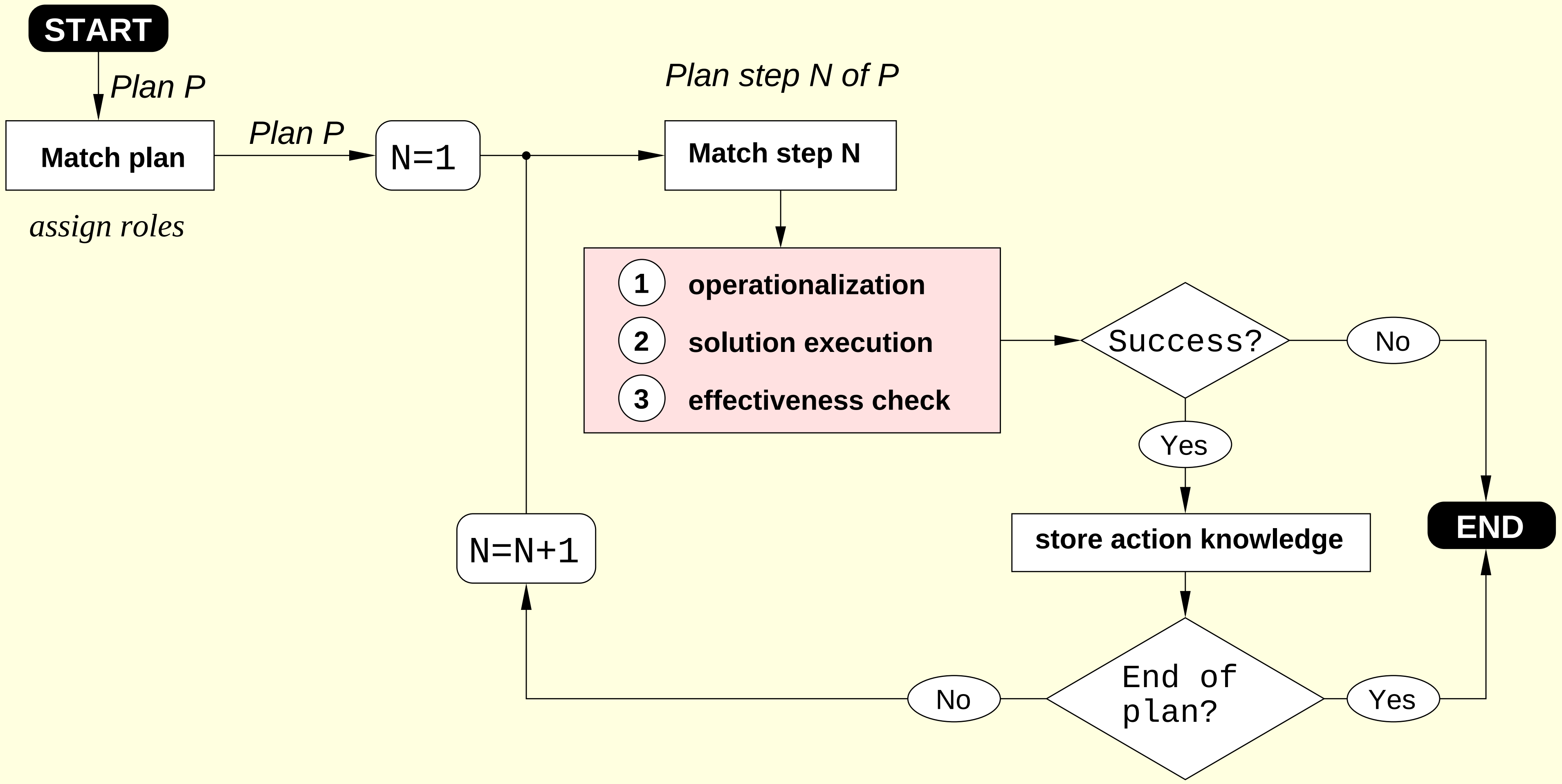
Methodology: Application

```
for an *operationalized* plan, P
    for an agent, A, that takes on role r in P
        for each step, s, of P
            – A identifies cases that can implement s in
              the current scenario using CBR
            – A selects one of these similar cases
              probabilistically
            – A executes the application knowledge in that
              retrieved case
            [RL Step]
            – A updates the success rate of the case
              based on the outcome
```

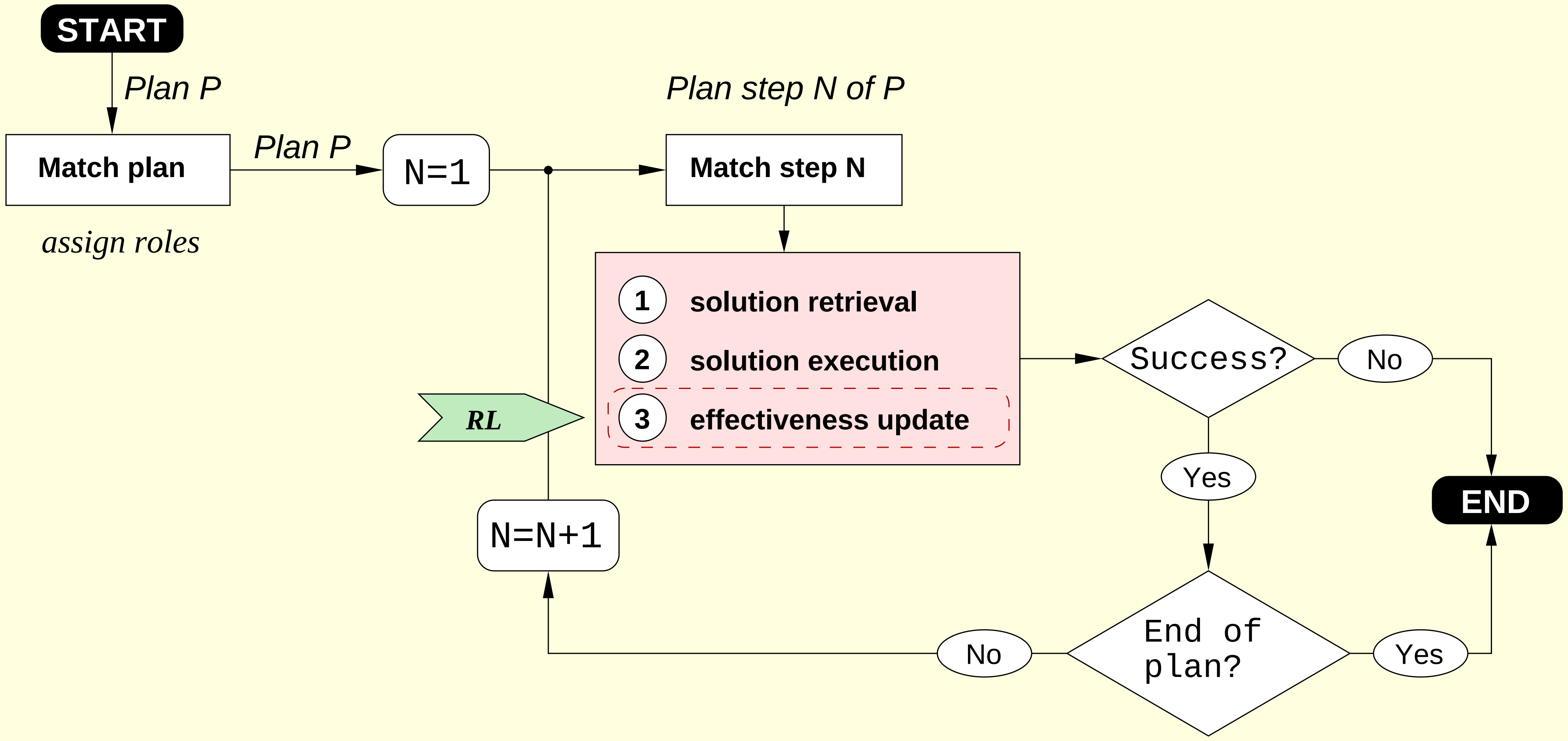
Methodology Summary


skeletal plans for a role
plans with non−reinforced application knowledge
a single merged plan with non−reinforced application knowledge
P_1
P_2
P_n
TRAINING
TRAINING
TRAINING
Plan Merging
APPLICATION (Retrieval)
APPLICATION (RL)
a single merged plan with non−reinforced application knowledge
a single merged plan with reinforced application knowledge

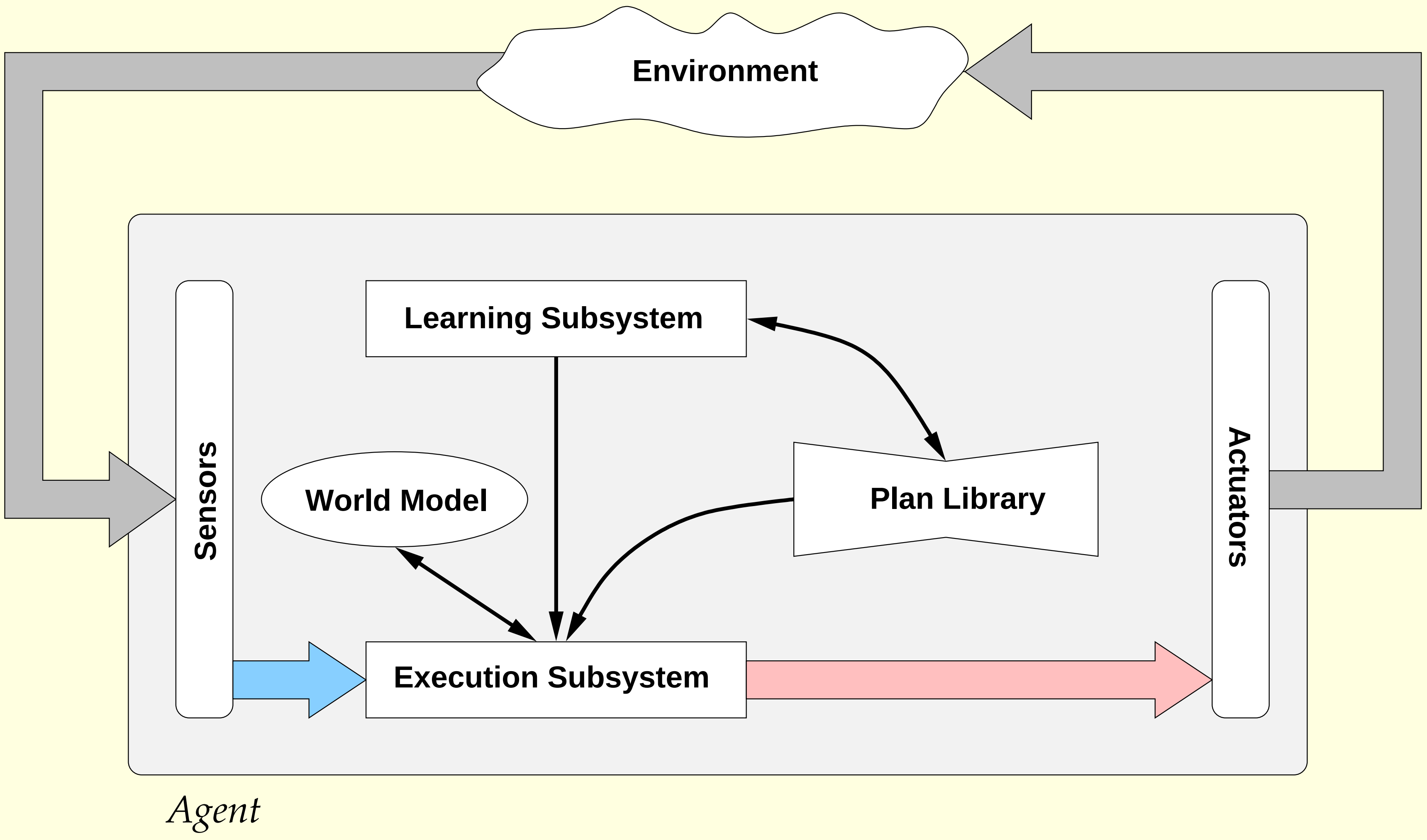
Methodology: Training



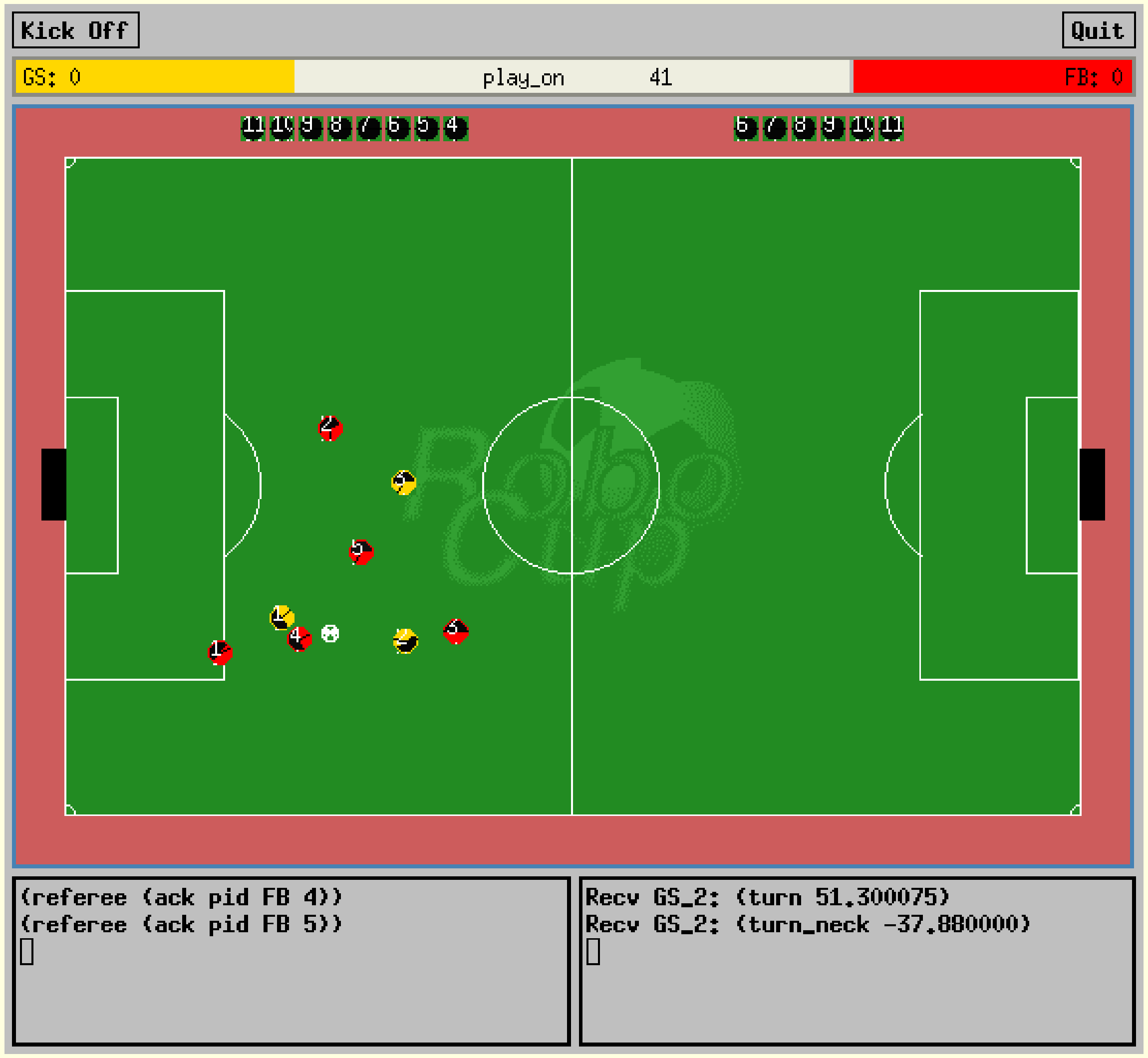
Methodology: Application



Agent Architecture



RoboCup Soccer Simulator



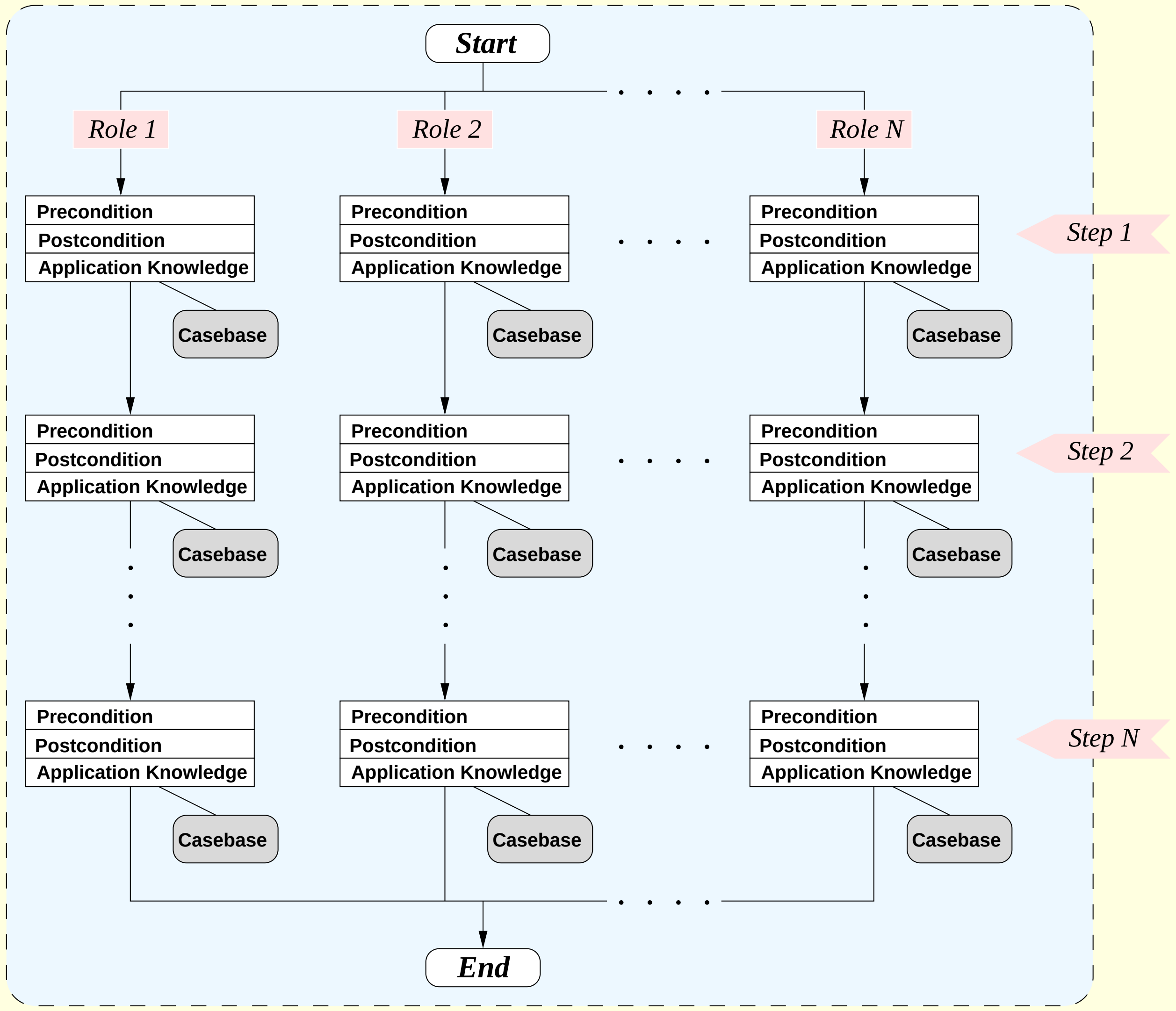
```
(referee (ack pid FB 4))
(referee (ack pid FB 5))
```

```
Recv GS_2: (turn 51.300075)
Recv GS_2: (turn_neck -37.880000)
```

Implementation: Plan Specification Language

```
(plan <symbol:plan-name>
   (rotation-limit <float[0..180]:max-rotation> <float[0..90]:rotation-increment>)
   (step <integer:plan-step-number>
      (precondition
         (timeout <integer:timeout-cycles>)
         (role <symbol:role-name> <integer:role-priority>
            ([not] <symbol:condition-name> <any:arg1> .. <any:argN>) .. )
         (role ...) .. )
      (postcondition
         (timeout <integer:timeout-cycles>)
         (role <symbol:role-name> -1 (<condition> <arguments>))
         (role ...) .. )
      (application-knowledge
         (case <symbol:plan-name> <symbol:role-name> <integer:plan-step-number>
            (gridunit <float:grid-unit>)
            (success <integer:success-count>) (failure <integer:failure-count>)
            (rotation <float:rotation>)
            (opponent-constellation (<integer:x-gridpos> <integer:y-gridpos>) .. (...))
            (action-sequence (<symbol:action-name> <any:arg1> .. <any:argN>) .. (...))
         )
         (case ...) .. ))
   (step ...) .. )
```

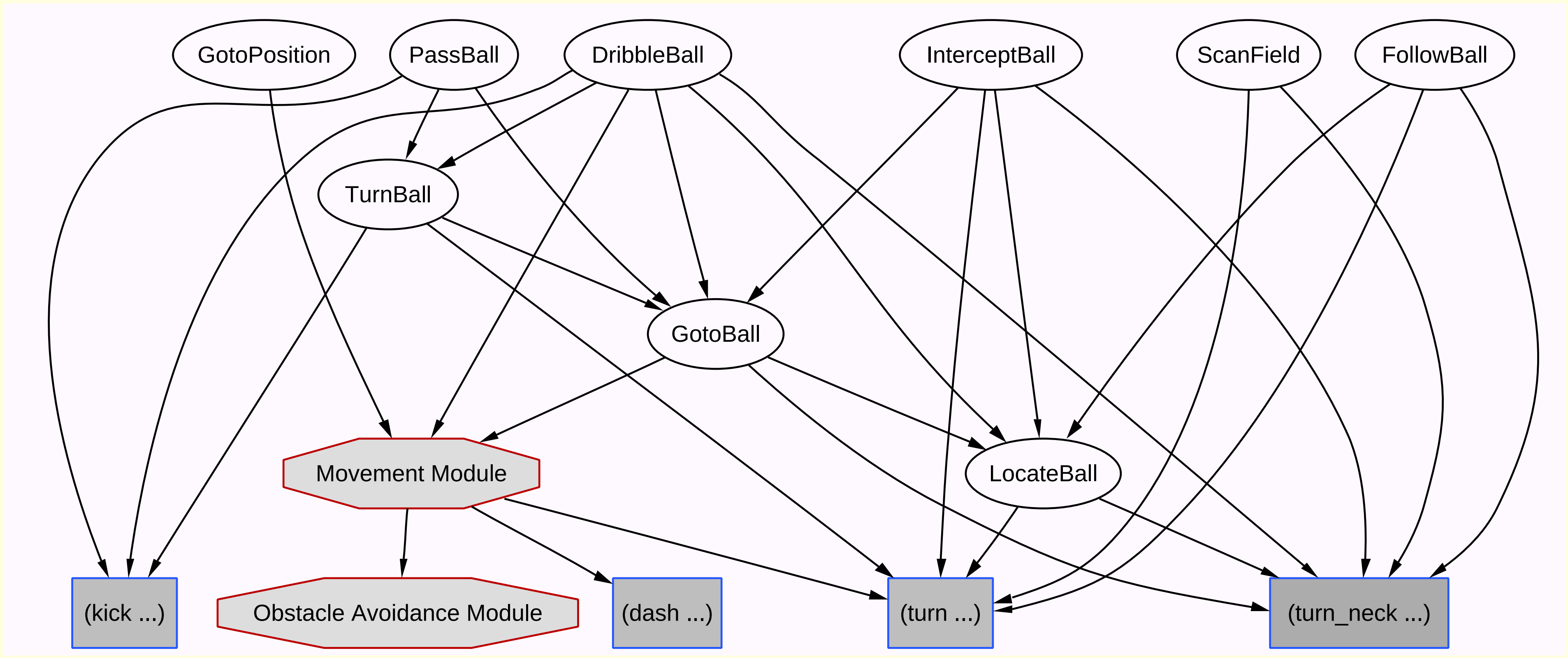
Implementation: Plan Data Structure



PLAN

Implementation: Reactive Action Modules

- Each reactive action module dynamically generates a sequence of primitive actions



Implementation: Programs

- Agent programs
  - soccerPlayer, soccerTrainer, dumbPlayer
  - 132 input parameters for customizing agent behavior
- Utilities
  - planstats, mergePlans, generateTestScenario
- Scripts
  - player, trainer, setup-train, setup-apply, runall, genresults, etc.

Implementation: Programs

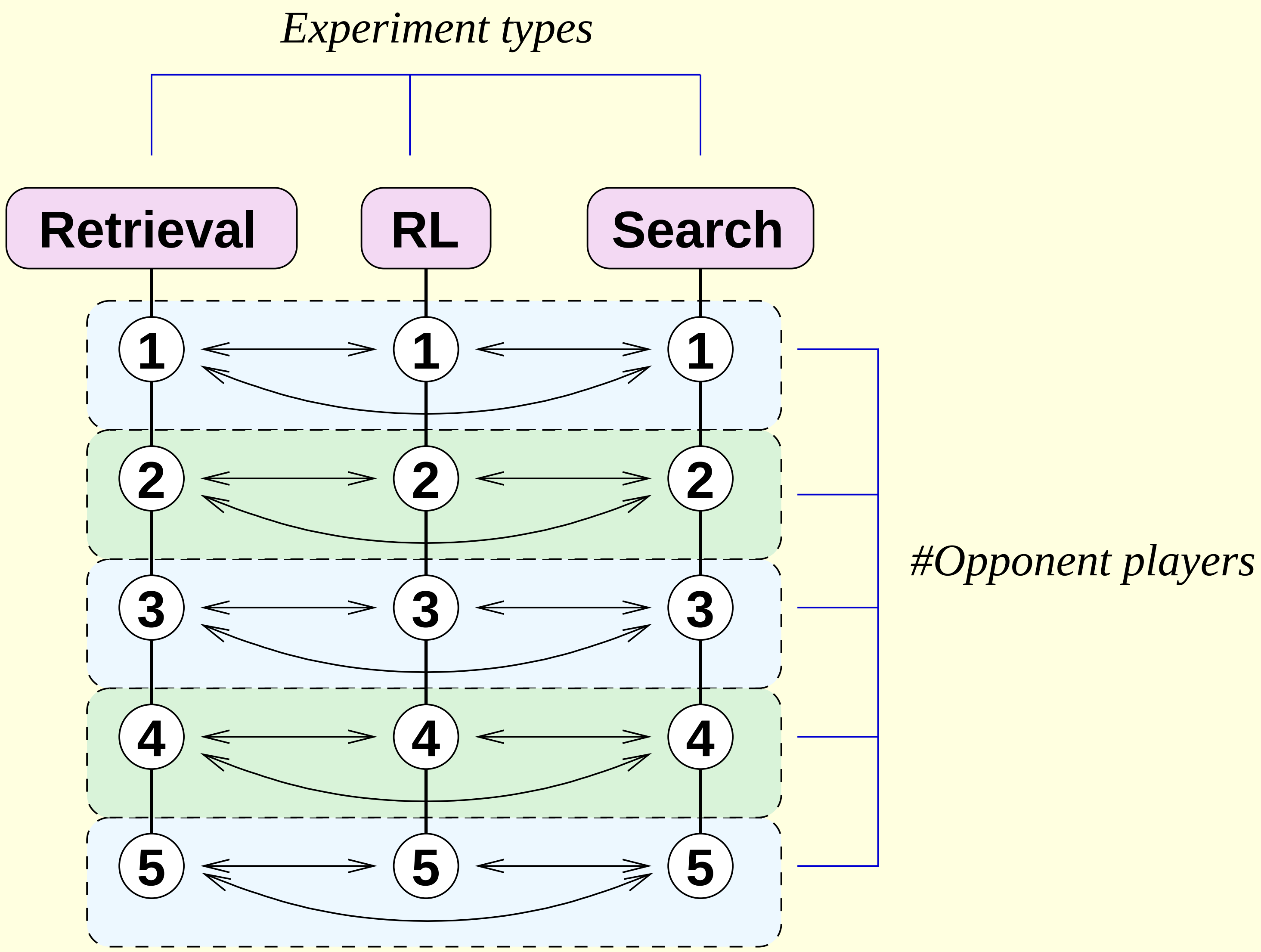
- ˜600 files
- ˜250 C++ classes
- ˜125,000 lines of code

Program Classes	*#Lines (approx.)*
Action	12,000
Search	14,000
Planning	25,000
Learning	2,000
Infrastructure	19,000
Debugging/Testing	25,000
Parsing	6,000
Agents	21,000

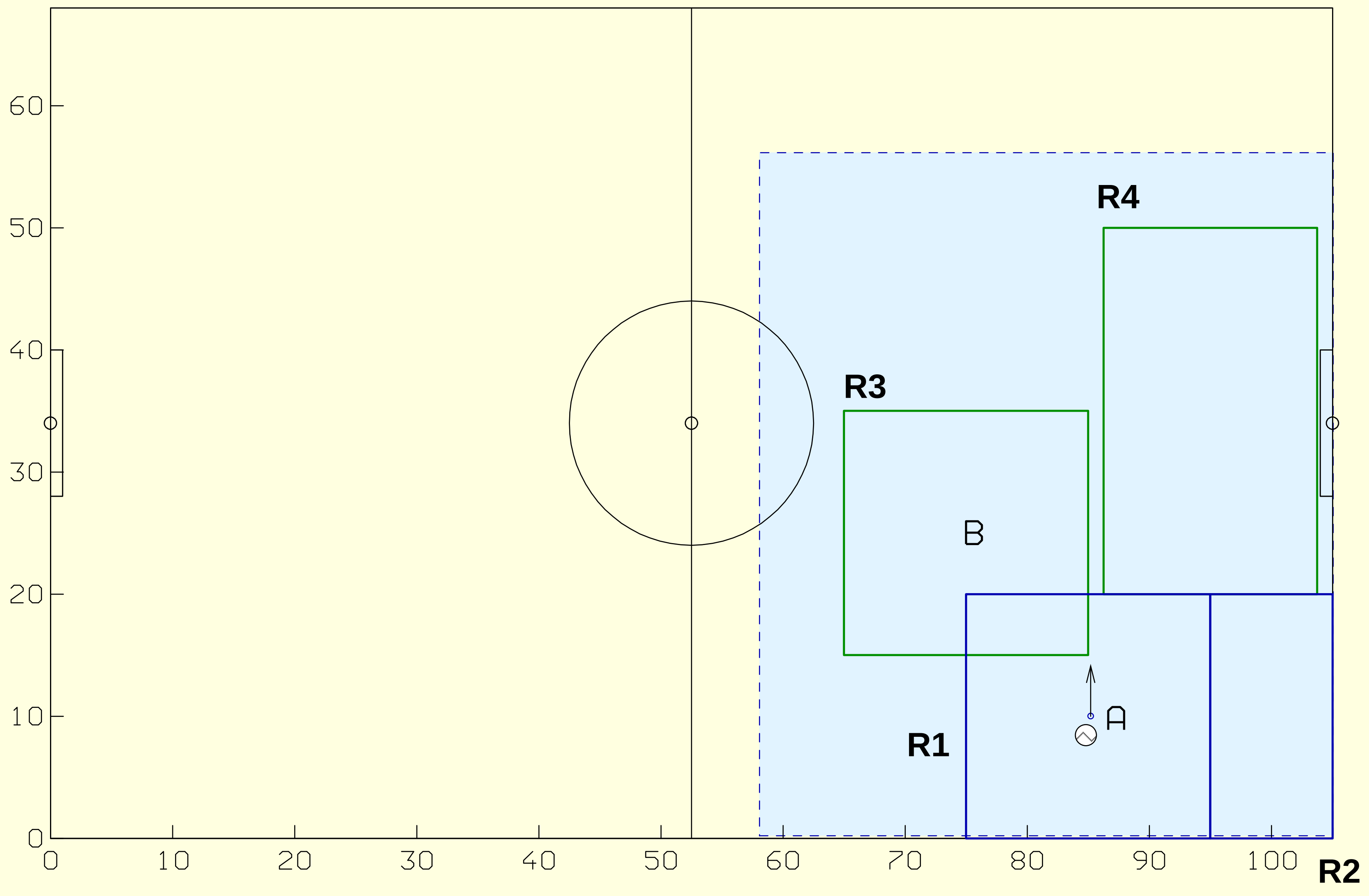
Evaluation Method

- Evaluation of multiagent systems is hard
- Experiments using different versions of the same program
  - Retrieval: each agent retrieves and applies its application knowledge
  - Search: each agent does search to implement its role
  - Naive RL: each agent retrieves, applies, and updates its application knowledge
- 4 test plans of varying complexity
- Statistical analysis using *paired t-test*

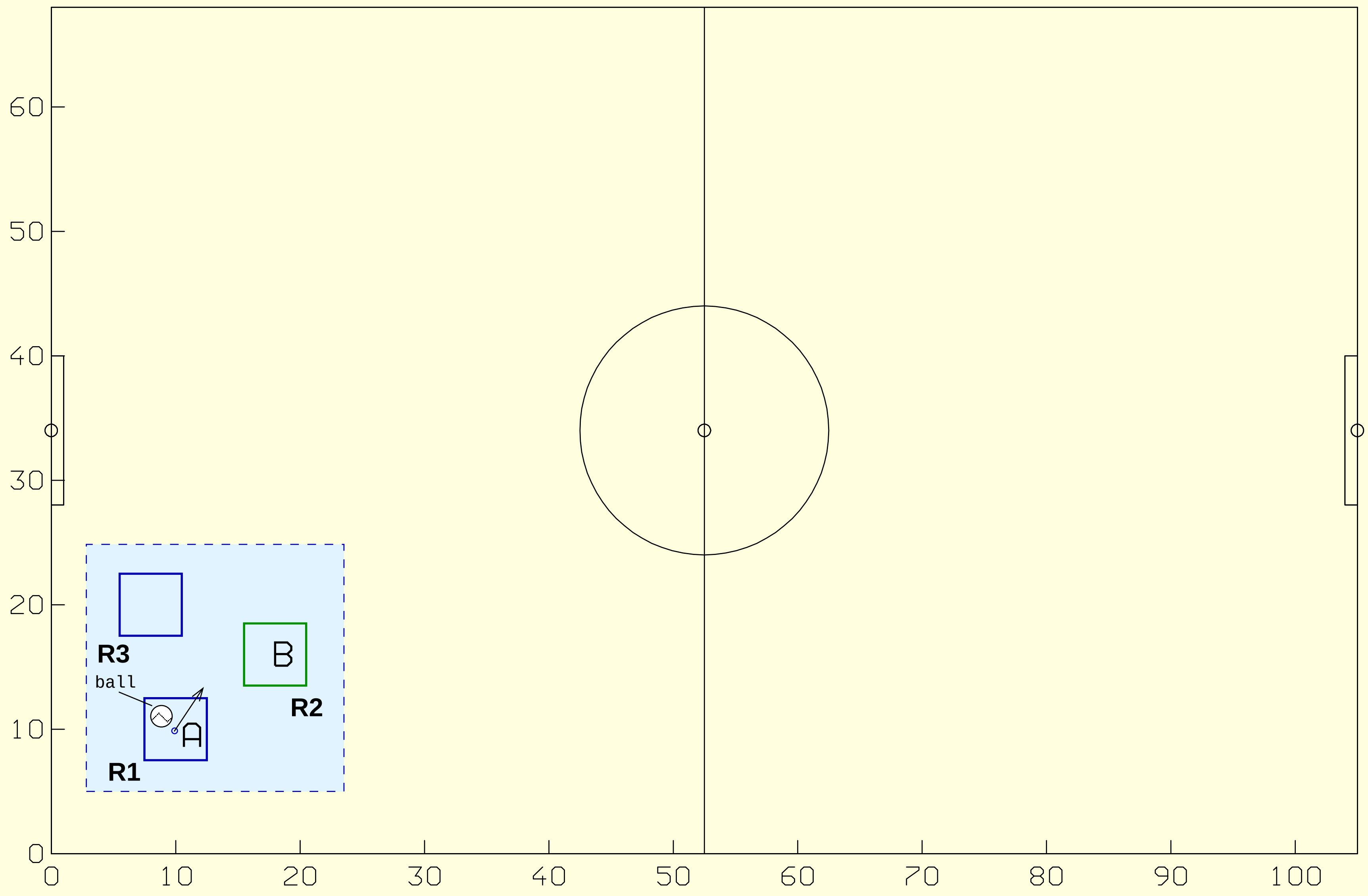
Evaluation Method (cont'd)



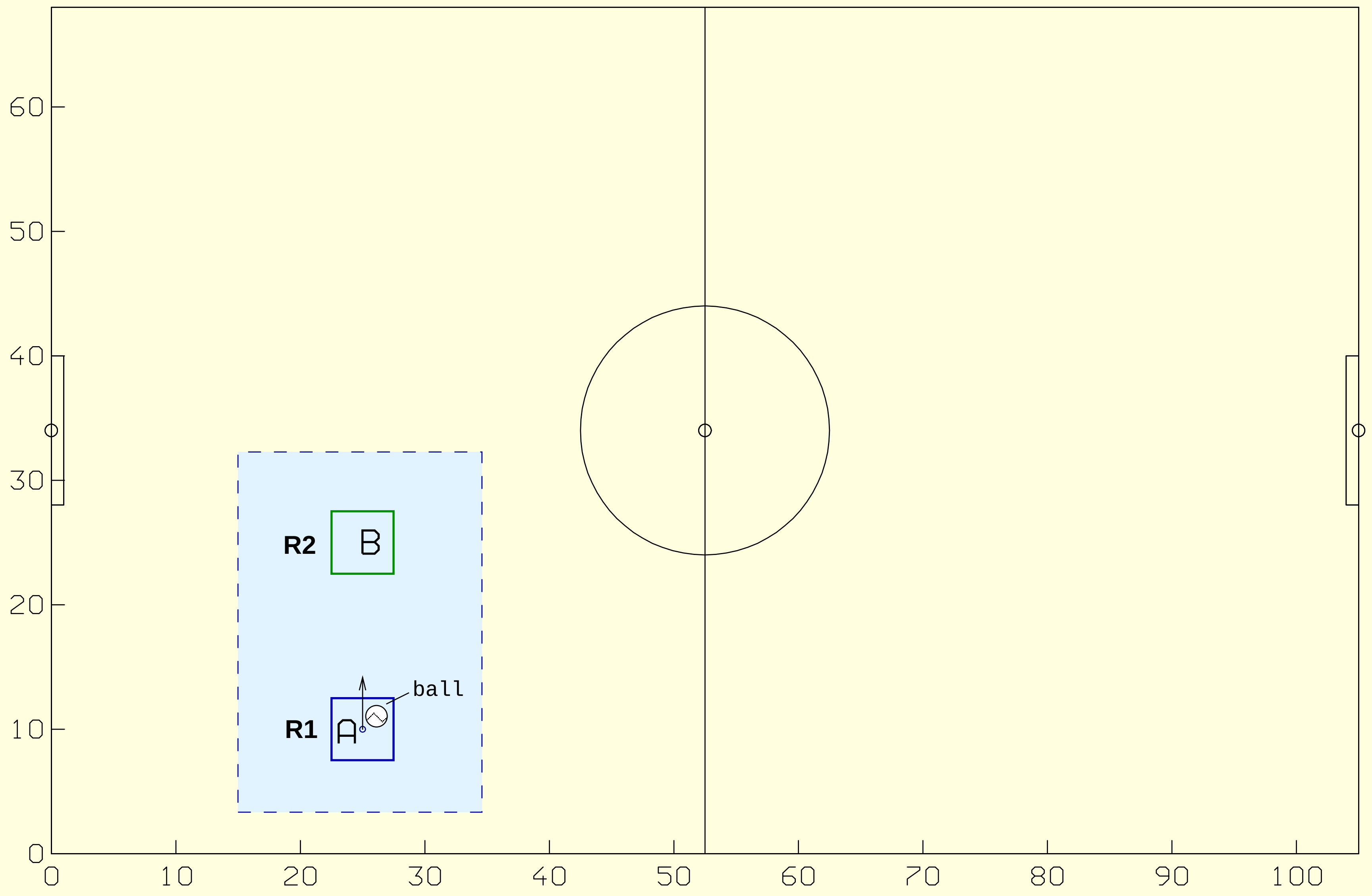
Test Plans: Centerpass (2 steps)



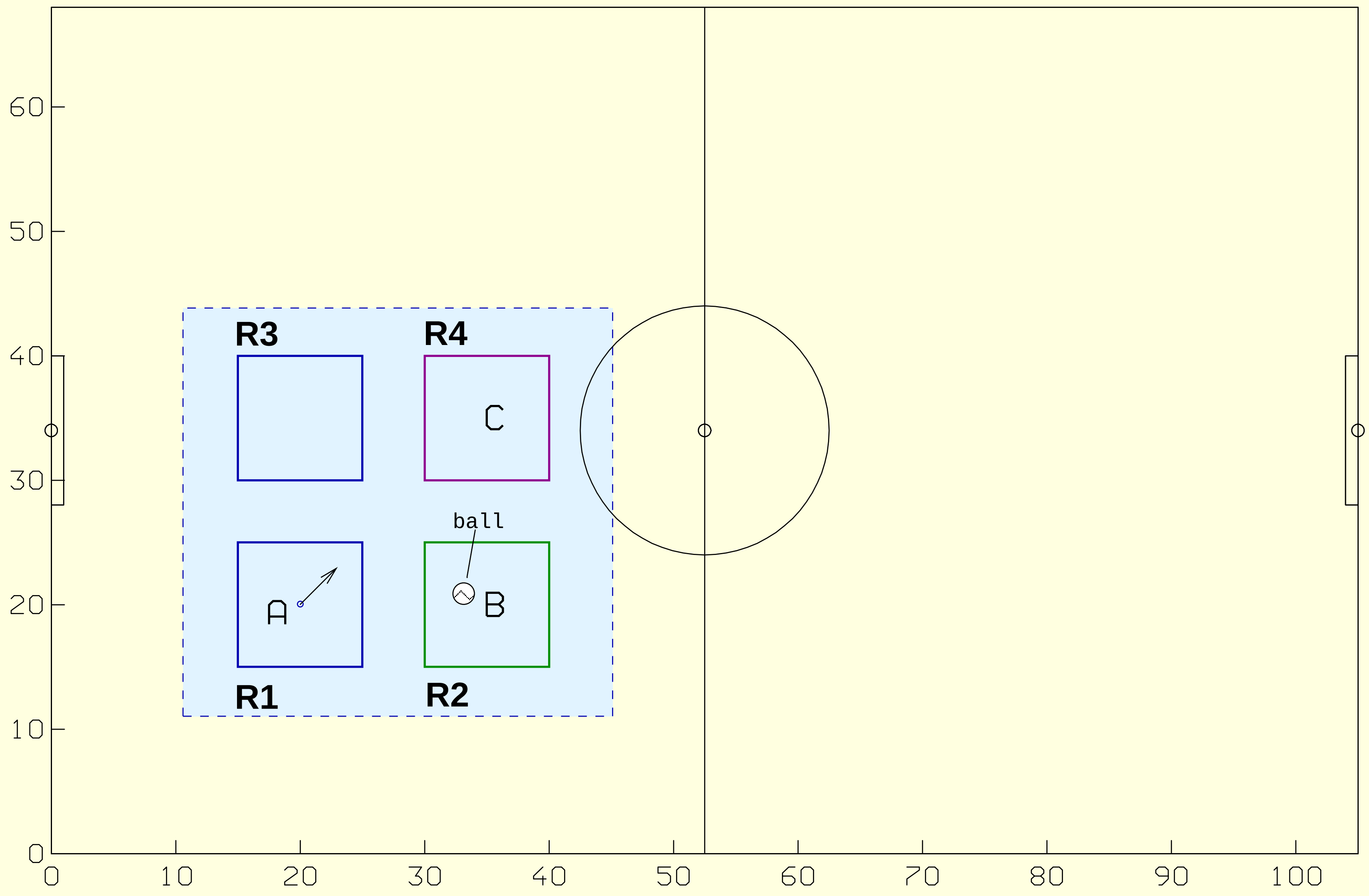
Test Plans: Givengo (3 steps)



Test Plans: Singlepass (1 step)



Test Plans: UEFA Fourwaypass (4 steps)



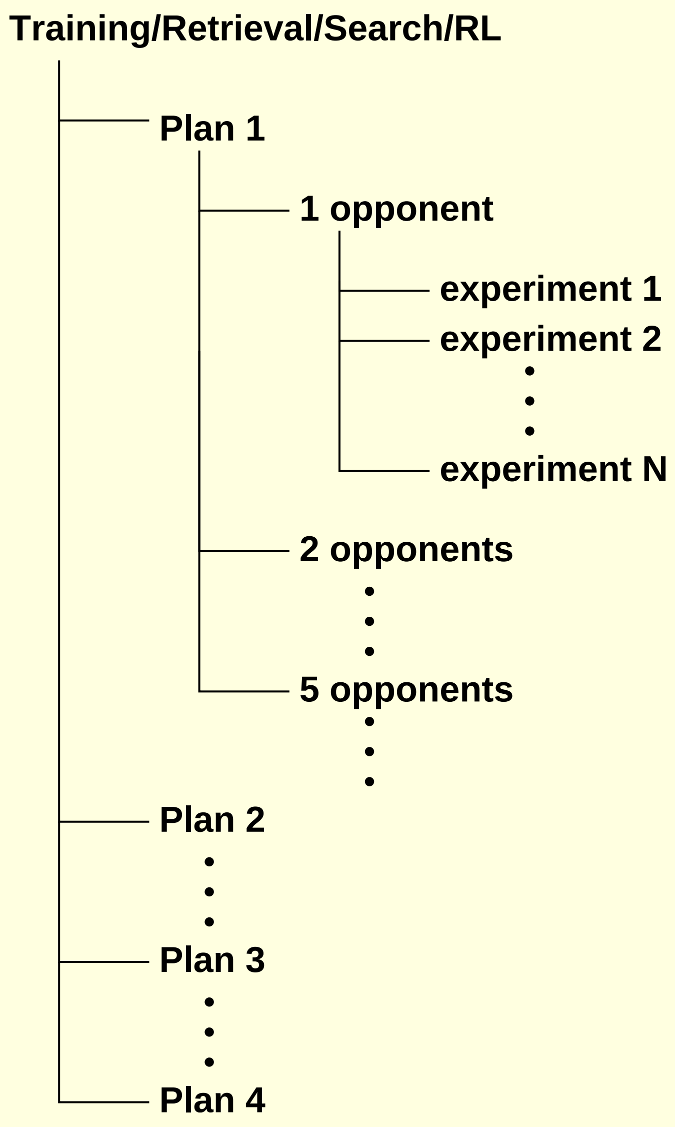
Experiment Setup

- Experiment and system modes

Experiments/Modes	RL	Search	Retrieval	Case Storage
TRAINING	**off**	**on**	**off**	**on**
APPLICATION (Retrieval)	**off**	**off**	**on**	**off**
APPLICATION (Learning)	**on**	**off**	**on**	**off**
APPLICATION (Search)	**off**	**on**	**off**	**off**

- Training: opponents do not move
- Application: opponents use a high-level ball interception behavior
- Plan skeletons are unchanged throughout testing

Experiment Setup Hierarchy



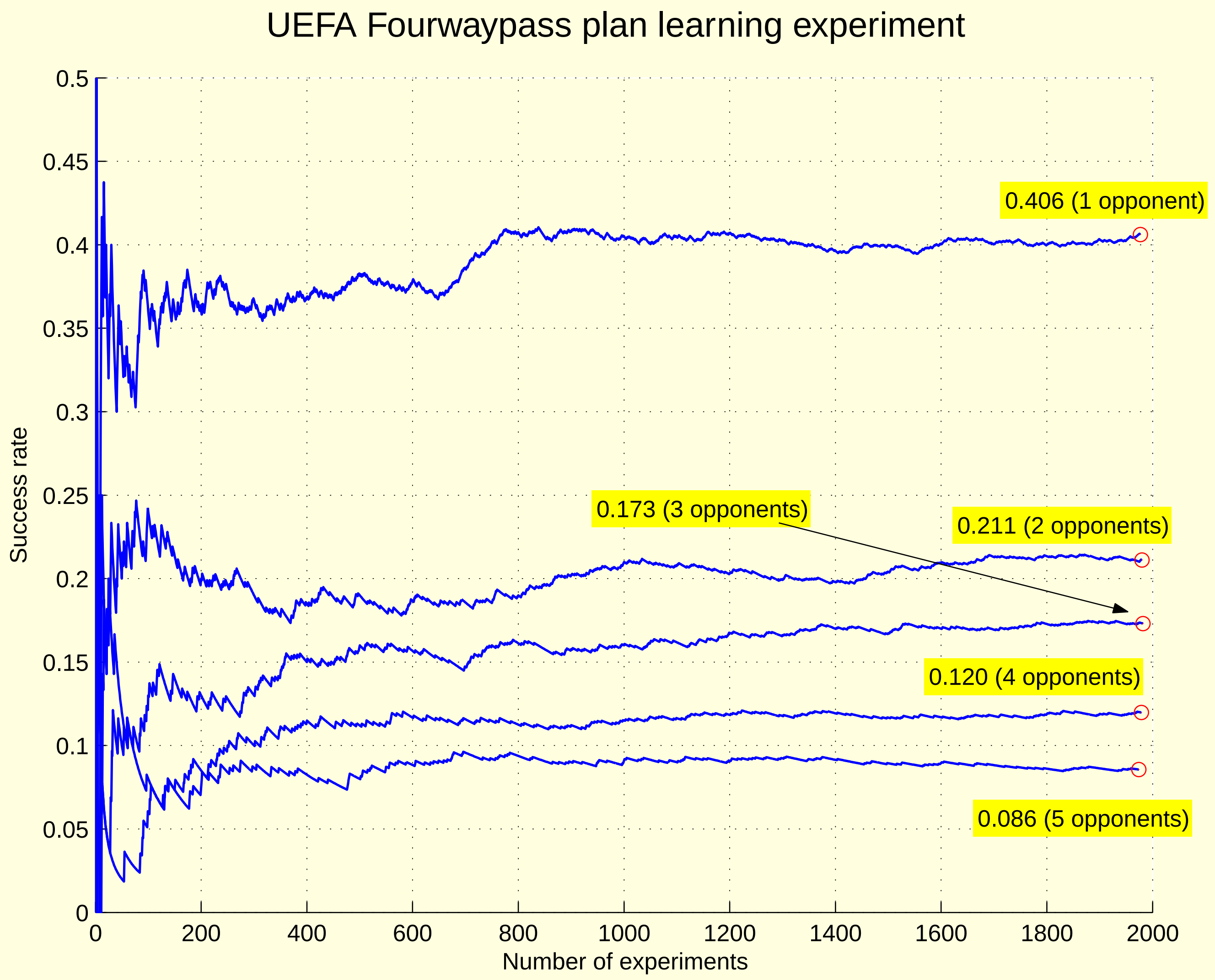
Experimental Results (Training)

```
Plan Centerpass: Roles A, B
    - Step 1 [2 roles with cases]
         - Role: A, #Cases: 1
         - Role: B, #Cases: 1
    - Step 2 [2 roles with cases]
         - Role: A, #Cases: 1
         - Role: B, #Cases: 1

Plan Givengo: Roles A, B
    - Step 1 [2 roles with cases]
         - Role: A, #Cases: 90
         - Role: B, #Cases: 1
    - Step 2 [1 role with cases]
         - Role: A, #Cases: 770
    - Step 3 [2 roles with cases]
         - Role: A, #Cases: 1
         - Role: B, #Cases: 9
```

Experimental Results (Training)

```
Plan Singlepass: Roles A, B
    - Step 1 [2 roles with cases]
         - Role: A, #Cases: 210
         - Role: B, #Cases: 1

Plan UEFA Fourwaypass: Roles A, B, C
    - Step 1 [2 roles with cases]
         - Role: A, #Cases: 1
         - Role: B, #Cases: 41
    - Step 2 [2 roles with cases]
         - Role: A, #Cases: 103
         - Role: C, #Cases: 1
    - Step 3 [1 role with cases]
         - Role: A, #Cases: 581
    - Step 4 [3 roles with cases]
         - Role: A, #Cases: 1
         - Role: C, #Cases: 2
```

Experimental Results (RL)



Experimental Results

Success rates of UEFA Fourwaypass plan experiments with $[1\ ..\ 5]$ opponents

	1	2	3	4	5
RL	0.402731	0.211927	0.173573	0.118803	0.085924
Retrieval	0.363636	0.194303	0.159960	0.114257	0.062753
Search	0.000000	0.001124	0.000000	0.000000	0.000000

The mean and standard deviation of the last 100 values from the UEFA Fourwaypass plan RL experiment

	1	2	3	4	5
mean	0.402731	0.211927	0.173573	0.118803	0.085924
std. dev.	0.001490	0.000829	0.000518	0.000587	0.000637

Paired t-test results for comparing UEFA Fourwaypass plan experiments

	p-value	t-value	confidence
RL vs Retrieval	0.026885	3.415755	95.0%
RL vs Search	0.023230	3.576998	95.0%
Retrieval vs Search	0.025039	3.493684	95.0%

UEFA Fourwaypass Experiment Summary

- Success rate decreases as #opponents increases
- Success rates in each opponent scenario converge
- Naive RL performs best, followed by Retrieval, and finally Search in all scenarios with high statistical significance (95%)

Comparison of Experimental Results

Singlepass (2 agents, 1 step)

	p-value	t-value	confidence
RL vs Retrieval	0.944	-0.075	–
RL vs Search	0.001	-10.216	99.9%
Retrieval vs Search	0.000	-12.376	99.9%

Centerpass (2 agents, 2 steps)

	p-value	t-value	confidence
RL vs Retrieval	0.120	1.974	85.0%
RL vs Search	0.006	5.212	99.0%
Retrieval vs Search	0.006	5.371	99.0%

Givengo (2 agents, 3 steps)

	p-value	t-value	confidence
RL vs Retrieval	0.050	2.772	90.0%
RL vs Search	0.001	9.421	99.9%
Retrieval vs Search	0.034	3.164	95.0%

UEFA Fourwaypass (3 agents, 4 steps)

	p-value	t-value	confidence
RL vs Retrieval	0.027	3.416	95.0%
RL vs Search	0.023	3.577	95.0%
Retrieval vs Search	0.025	3.494	95.0%

Comparison Summary

- Success rates decrease as #opponents increases in all test plans

- In Singlepass plan
  - Search performs significantly better than both RL and Retrieval due to the simplicity of the scenario
  - RL and Retrieval performances are statistically identical likely due to inherent lack of variability in the solutions needed

Comparison Summary (cont'd)

- RL performs better than Retrieval and Search, and Retrieval performs better than Search in Centerpass, Givengo, and UEFA Fourwaypass test plans
- RL performs increasingly better than Retrieval as plan complexity increases (85%, 90%, 95%)
- In simpler plans, Search can find highly successful solutions. In harder plans, Search fails to find any solutions in almost all scenarios likely due to the lack of feedback that RL mode has

Example Runs

(a) Fourwaypass, RL [+]

(b) Fourwaypass, Retrieval [+]

(c) Fourwaypass, Search [-]

(d) Givengo, Retrieval [+]

(e) Givengo, Search [+]

(f) Centerpass, RL [-]

Conclusions

- Experimental results show that the MuRAL methodology can enable a group of goal-directed autonomous agents with shared goals to behave collaboratively and coherently in complex, dynamic, and uncertain domains

- MuRAL uses high-level plans to implement a *learning-by-doing* solution to multiagent problems
  - via case-based learning during training and
  - via naive reinforcement learning during application

Contributions

- MuRAL: A *learning-by-doing* solution methodology for enabling agents to learn to apply high-level plans to dynamic, complex, and uncertain environments
- An agent architecture that enables reactive and collaborative strategy selection and application in situated environments
- A plan application approach that combines case-based reasoning and naive reinforcement learning
- A symbolic representation method for specifying both high-level multiagent and single-agent plans and for storing application knowledge in the form of cases

Contributions (cont'd)

- An unsupervised learning algorithm that incorporates case-based learning and naive reinforcement learning
- A fully-implemented MuRAL system that works in the RoboCup soccer simulator

Limitations

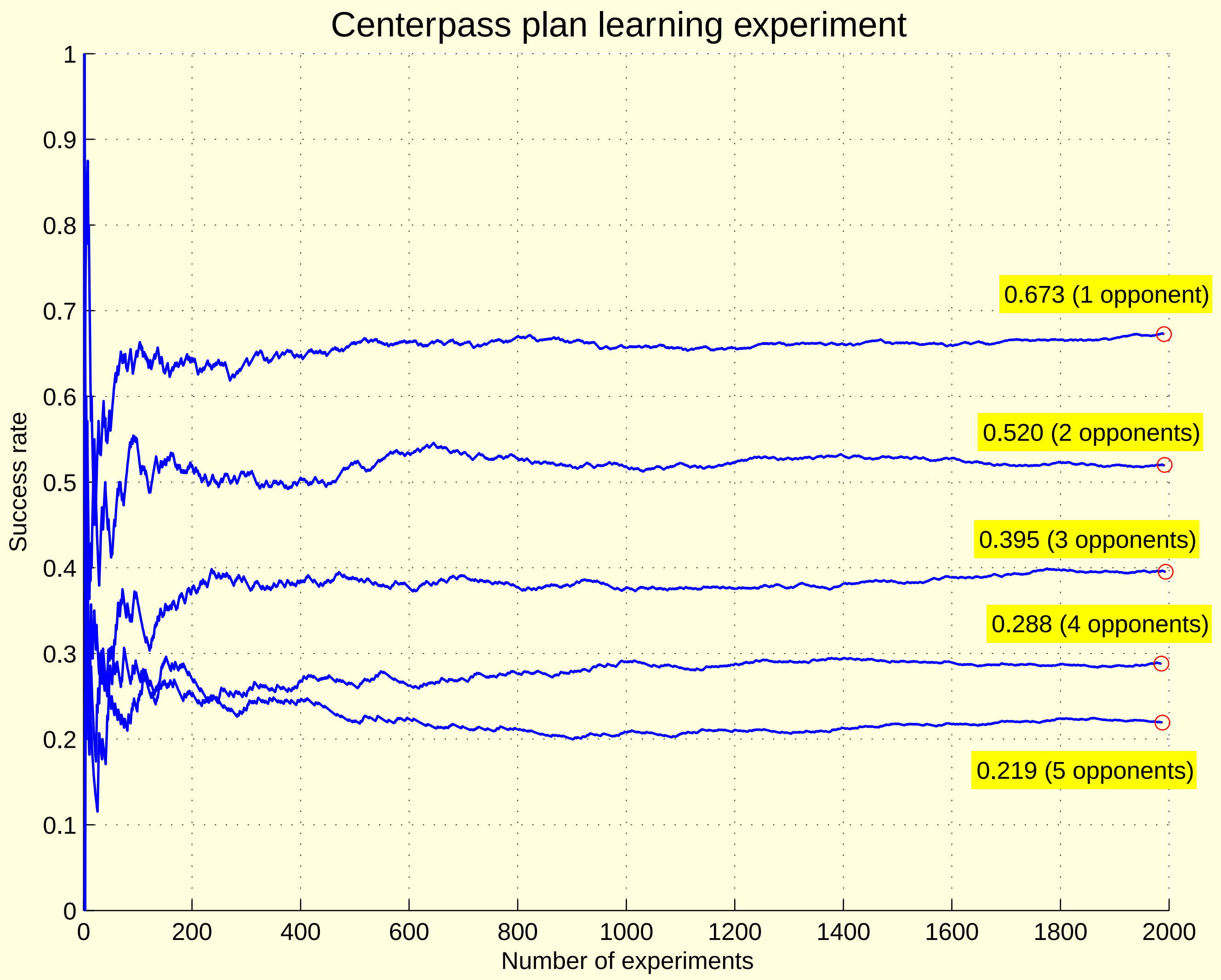
- Skeletal plans need to be manually written
- No learning of skeletal plans

Future Work

- Automated recognition of learning opportunities arising from failures, either due to lack of plans or plan implementations
- Coherent successive plan application
  - How to select plans consistently
  - How to assign roles consistently
  - How to coordinate behavior
- Other issues
  - Resource allocation
  - Dynamic decision making for termination of failed plans instead of timeouts?

◇

Experimental Results (Learning)



Experimental Results (Learning)

Success rates of Centerpass plan experiments with $[1\ ..\ 5]$ opponents

	1	2	3	4	5
RL	0.670781	0.517448	0.395180	0.286241	0.221279
Retrieval	0.625626	0.509018	0.347041	0.288577	0.215726
Search	0.042169	0.043000	0.018036	0.013026	0.018090

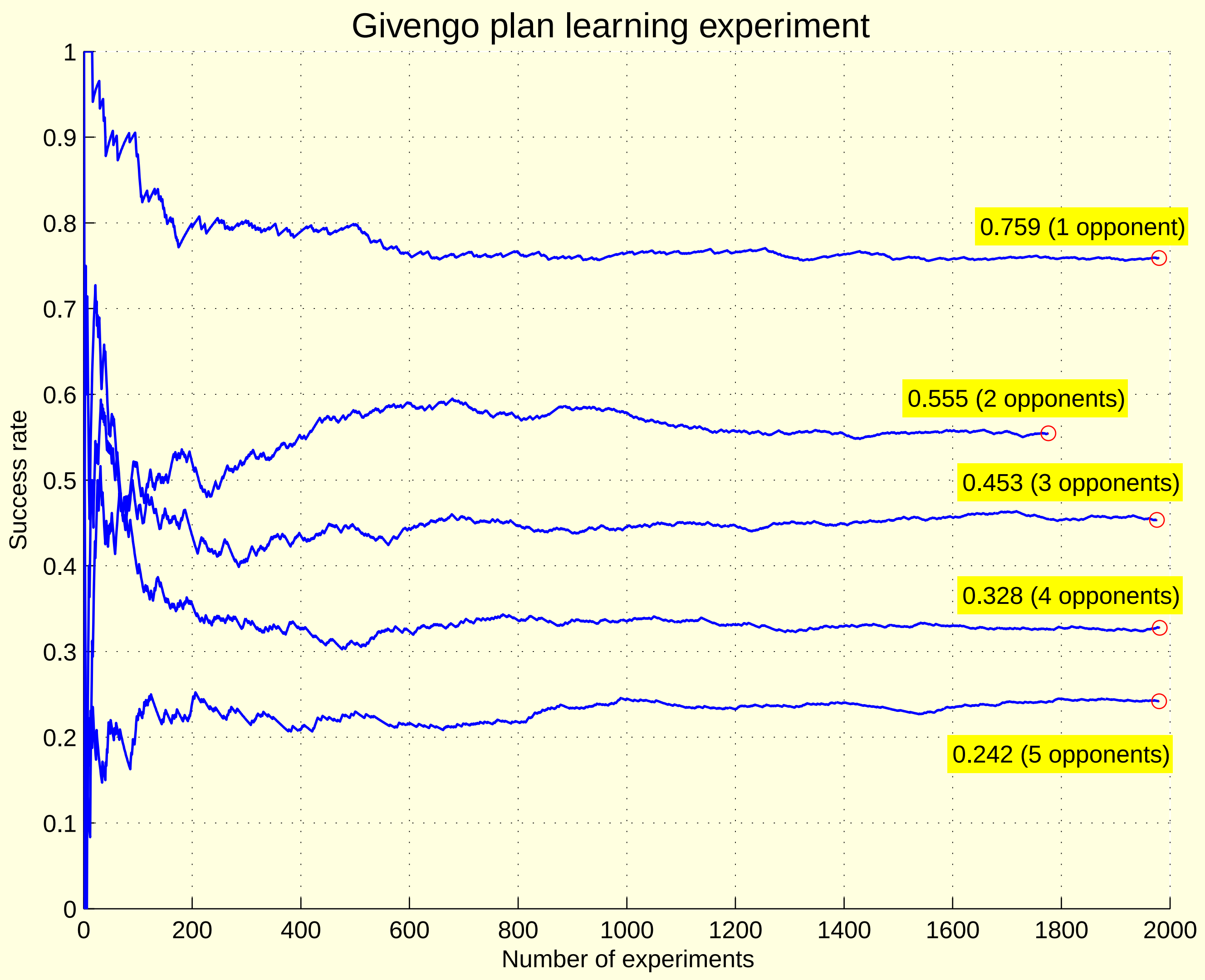
The mean and standard deviation of the last 100 values from the Centerpass plan RL experiment

	1	2	3	4	5
mean	0.670781	0.517448	0.395180	0.286241	0.221279
std. dev.	0.001665	0.003728	0.000805	0.001383	0.000843

Paired t-test results for comparing Centerpass plan experiments

	p-value	t-value	confidence
RL vs Retrieval	0.119635	1.973892	85.0%
RL vs Search	0.006465	5.211575	99.0%
Retrieval vs Search	0.005805	5.370713	99.0%

Experimental Results (Learning)



Experimental Results (Learning)

Success rates of Givengo plan experiments with $[1 \ .. \ 5]$ opponents in RL, Retrieval and Search modes

	1	2	3	4	5
RL	0.758004	0.553951	0.456083	0.325450	0.242873
Retrieval	0.740628	0.543611	0.381874	0.306061	0.204040
Search	0.672556	0.491210	0.370291	0.257848	0.196262

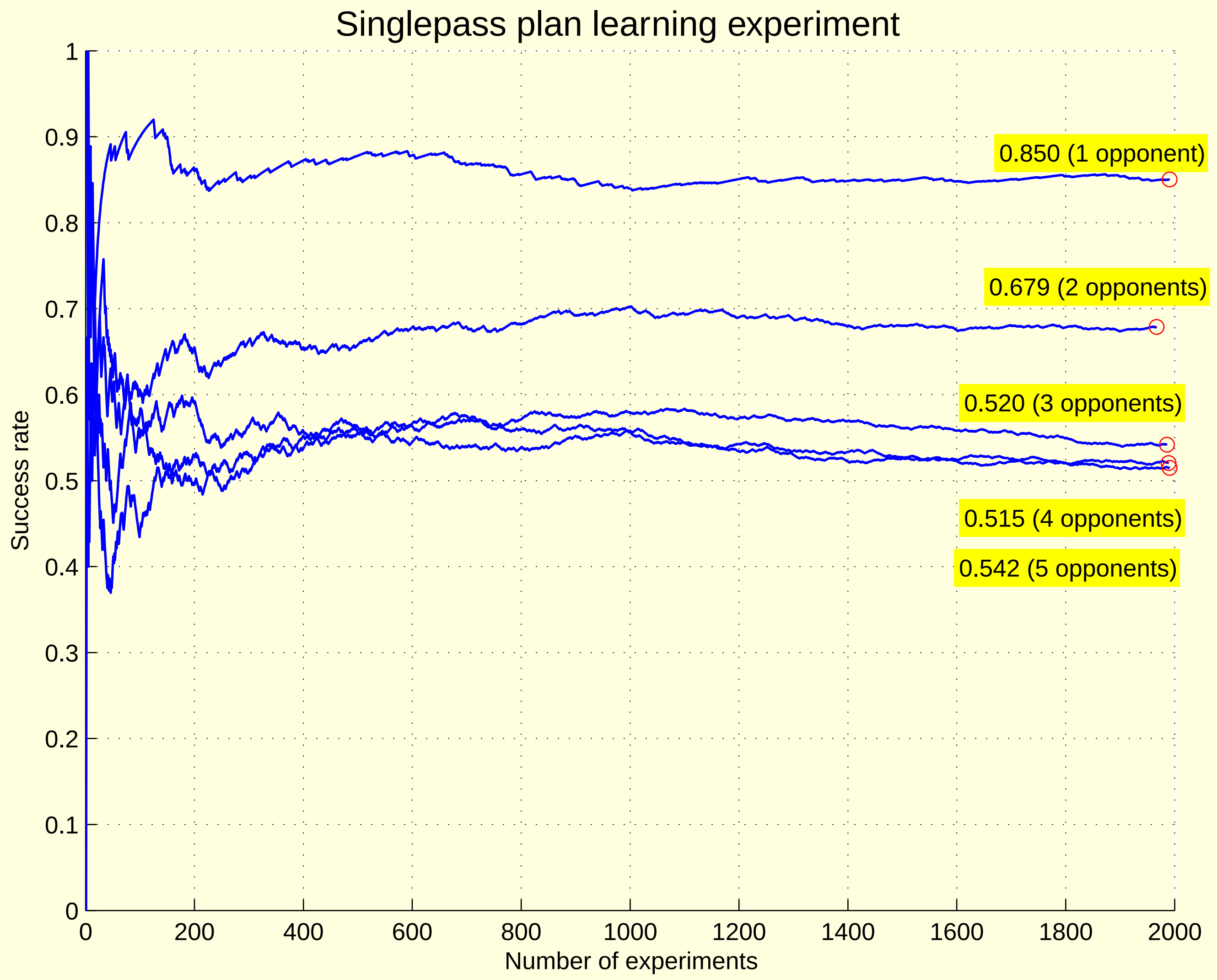
The mean and standard deviation of the last 100 values from the Givengo plan RL experiment

	1	2	3	4	5
mean	0.758004	0.553951	0.456083	0.325450	0.242873
std. dev.	0.000913	0.001630	0.001339	0.001096	0.000764

Paired t-test results for comparing Givengo plan experiments

	p-value	t-value	confidence
RL vs Retrieval	0.050232	2.771919	90.0%
RL vs Search	0.000708	9.420699	99.9%
Retrieval vs Search	0.034065	3.163648	95.0%

Experimental Results (Learning)



Experimental Results (Learning)

Success rates of Singlepass plan experiments with $[1 \, .. \, 5]$ opponents

	1	2	3	4	5
RL	0.851200	0.676333	0.521230	0.514600	0.541750
Retrieval	0.835836	0.674000	0.508526	0.538616	0.550856
Search	0.901000	0.757545	0.602603	0.600200	0.637550

The mean and standard deviation of the last 100 values from the Singlepass plan RL experiment

	1	2	3	4	5
mean	0.851200	0.676333	0.521230	0.514600	0.541750
std. dev.	0.001813	0.001258	0.001268	0.000535	0.000909

Paired t-test results for comparing Singlepass plan experiments run in RL, Retrieval, and Search modes

	p-value	t-value	confidence
RL vs Retrieval	0.944030	-0.074713	–
RL vs Search	0.000517	-10.215832	99.9%
Retrieval vs Search	0.000245	-12.375628	99.9%