systems[2].

Prior to conducting the research reported in this chapter, we used neural networks to teach an agent to shoot a moving ball into a goal in a simulator based closely upon the Dynasim system [Sahota 96]. The simulator models the non-holonomic wheeled Dynamite robots [Sahota *et al.* 95]. In this scenario, we were able to use one neural network to time the agent's approach towards the ball, and another to aim the ball. The learned behavior successfully enabled the agent to redirect a moving ball with varying speeds and trajectories into specific parts of the goal. By carefully choosing the input representation to the neural networks so that they would generalize as much as possible, the agent was able to use the learned behavior in all quadrants of the field even though it was trained in a single quadrant [Stone and Veloso 98d].

Learning was applied in both simulated and real robotic, to hit a stationary ball into the goal using the concept of learning from easy missions [Asada *et al.* 94a]. By first placing the ball directly in front of the goal and then gradually moving it farther away, thus making the task harder, the learning time was reduced from exponential to linear in the size of the state space within an RL framework. The robots achieved 70% success in simulation and 40% success in the real world.

Another early example of learning an individual skill in the RoboCup soccer server involved using a neural network to enable an agent to learn when to shoot and when to pass [Matsubara *et al.* 96]. An agent was given the ball near the opponent's goal with a goaltender blocking the way and a teammate nearby. Based on the positions of the ball, the goaltender, and the teammate, the agent with the ball learned when it was best to shoot directly at the goal and when it was best to pass.

In a similar setup, using the same Dynasim-based simulator mentioned above, we used memory-based learning to allow an agent to learn when to shoot and when to pass the ball [Stone and Veloso 96a]. This learning was of an individual skill since the goaltender and the teammate used deterministic, fixed behaviors.

[2] A full discussion of related robotic soccer systems, including more recent learning approaches, appears in Chapter 9 (Section 9.2).

Chapter 10

Conclusions and Future Work

Motivated by the challenges inherent in the soccer server, a simulated robotic soccer domain, this thesis contributes several techniques for building successful agents in real-time, noisy, collaborative and adversarial environments. This chapter reviews the dissertation's scientific contributions to the fields of Multi-Agent Systems and Machine Learning (Section 10.1) and then discusses promising directions for future research in this challenging class of domain (Section 10.2).

10.1 Contributions

The four main contributions of this thesis are summarized as follows.

- The **team member agent architecture** presented in Chapter 3 is suitable for PTS domains: domains with team synchronization opportunities interleaved with periods of real-time, dynamic action and limited communication. The team member agent architecture incorporates a locker-room agreement, which includes the definition of a flexible teamwork structure including mechanisms for task decomposition and dynamic role assignment. Roles, team formations, and multi-agent set-plays are defined.

  For PTS domains with limited communication (as opposed to no communication) possible during the dynamic periods, the locker-room agreement also defines inter-agent communication protocols. This thesis presents a communication paradigm for PTS domains with single-channel, low-bandwidth, unreliable communication environments. This communication paradigm facilitates team coordination while being robust to lost messages and active interference from opponents.

  The team member agent architecture is implemented both in simulated robotic soccer and in real robotic soccer. Empirical results demonstrate the effectiveness of the teamwork structure and communication paradigm implementations.

- **Layered learning** (Chapter 4) is a hierarchical ML paradigm applicable in complex domains in which learning a mapping directly from sensors to actuators is intractable. Given a hierarchical task decomposition, layered learning allows for learning at each level of the hierarchy, with learning at each level directly affecting learning at the next higher level.

Layered learning is applied in simulated robotic soccer as a set of three interconnecting learned behaviors. At the lowest layer, ball interception, an individual skill, is learned (Chapter 5). This learned individual skill is used as part of the training behavior for learning pass evaluation, a multi-agent behavior (Chapter 6). The learned multi-agent behavior is then used to create the input space for learning pass selection, a team behavior (Chapter 7). All of the learned behaviors are validated empirically in controlled testing scenarios.

- **TPOT-RL** (Chapter 7) is a new multi-agent reinforcement learning method. TPOT-RL is designed for complex domains in which agents cannot necessarily observe the state transitions caused by their or other agents' actions. It exploits local, action-dependent features to aggressively generalize its input representation for learning and partitions the task among the agents, allowing them to simultaneously learn collaborative policies by observing the long-term effects of their actions. TPOT-RL is applicable in domains with large state spaces and limited training opportunities.

  TPOT-RL is developed and tested in the simulated robotic soccer domain. It is also successfully applied in another multi-agent domain, namely network routing.

- The **CMUnited simulated robotic soccer system** (Chapters 3–8 and Appendices B–D) is a fully implemented and operational team of simulated robotic soccer agents which has performed successfully at international tournaments. This thesis contributes algorithmic details of the CMUnited implementation, as well as its source code.

10.2 Future Directions

10.2.1 Future Directions within Robotic Soccer

The challenges inherent in the simulated robotic soccer domain have motivated the research reported in this dissertation. While CMUnited has performed successfully in competitions as a result of this research, there are still many challenges within robotic soccer to be addressed. These challenges could be addressed either within robotic soccer or within a similar real-time, collaborative, and adversarial domain.

Short-term, on-line learning and prediction. The on-line learning implemented in this thesis—TPOT-RL—takes many games against a fixed opponent to learn an effective behavior. It is still an open question whether useful on-line learning can be accomplished in this domain within the course of a single 10-minute game. For example, can the actions of a particular opponent agent be learned quickly enough that predictions can be exploited before the end of the game?

Opponent modeling. All of the robotic soccer behaviors created in this thesis have either ignored opponents' behaviors or considered them a part of the environment. Explicit modeling of opponents' behaviors and short-term goals may facilitate the adaptation of behaviors to specific adversarial situations.

On-line multi-agent planning. The set-plays created as part of our flexible teamwork structure are pre-compiled multi-agent plans. They are created ahead of time for use in situations that occur repeatedly within the domain. The creation of multi-agent plans on-line for use during an unanticipated situation remains an open research problem.

Since the robotic soccer simulator changes periodically, adding new challenges to replace those that have been successfully met, I anticipate additional research challenges from this domain in the future. For example, if the soccer server moves from a 2-dimensional to a 3-dimensional simulation, or if the visual information format is changed to simulate video camera input, the low-level skills will become more challenging to create, thus introducing additional learning opportunities. In addition, if heterogeneous agents or a coach agent are introduced into the environment, the resulting collaborative and adversarial aspects of the domain will present new research challenges.

This dissertation contributes several successful techniques for building agents in the robotic soccer domain. However, it is not yet a "solved" domain.

10.2.2 Future Directions within the Contributed Algorithms

In this section I discuss future research directions regarding the main contributions of this thesis: the team member agent architecture, layered learning, and TPOT-RL. Each contribution has been implemented in simulated robotic soccer and at most one other domain. In all three cases, an important future direction is the application of these algorithms to additional domains. Broad applications can verify the algorithms' generality as well as helping to identify their most important features for achieving successful results.

Team Member Agent Architecture

Automatic set-play definition. The flexible teamwork structure specifies how to execute pre-compiled, multi-agent plans, or *set-plays*. However, these set-plays must currently be input to the system. A general mechanism for automatically defining set-plays would be a useful direction for future research.

One possible approach to automatically defining set-plays is to use a probabilistic planning algorithm such as Weaver [Blythe 98]. Since PTS domains are dynamic, a probabilistic planner is needed to deal with unpredictable environmental changes. In noisy domains, the probabilistic methods are also needed to handle the uncertain effects of agents' actuators.

In order to use such a planning algorithm, one must define an *initial state*, a *goal state*, and a set of *operators*. The initial state is the situation in which the set-play is called, the goal state is the team's joint goal G (or an intermediate state on a path of states towards G), and the operators are the agents' actuators in the domain. Since set-plays are defined as part of the locker-room agreement, the planning algorithm could create a set-play for the entire team at once. The team could then break the plan into its individual roles in order to encode it within the locker-room agreement.

When to switch roles. The flexible teamwork structure defined within the team member agent architecture defines a mechanism by which agents can smoothly switch roles within a formation. However, we have not discovered any useful heuristics governing *when* agents should switch roles. Defining such heuristics would be an interesting extension to our teamwork structure.

For example, if agent a_1 is filling role r_1 and agent a_2 is filling role r_2, one clear reason to switch roles would be if a_1 has more of the resources necessary for filling r_2 than does a_2 and likewise a_2 for r_1. In this and other situations in which homogeneous agents might switch roles, the cost to the team in terms of possible discontinuities of roles being filled and in terms of inconsistent agent impressions of the mapping $A \mapsto R$ must be considered.

In addition to being implemented in the simulated robotic soccer domain, the team member agent architecture, including the flexible teamwork structure, has been applied in the real robotic soccer domain. Since our robots do not use inter-agent communication, the communication paradigm for PTS domains with single-channel, low-bandwidth, unreliable communication environments does not apply on our current robots. For soccer-playing robots with inter-agent communication capabilities, however, the communication paradigm can be applied.

An important direction of future work is to apply the team member agent architecture, the teamwork structure, and the communication paradigm to domains other than robotic soccer. The most obvious extension is to other team-sports domains, such as American football. As opposed to soccer, football has much more frequent synchronization opportunities: between every play, teammates can discuss their next play. Even so, formations can change during the course of a single play. And like in soccer, each formation consists of a set of roles that are each defined with some flexibility left to the player that fills it. The communication environment during the dynamic periods is also quite similar to soccer: it is single-channel, low-bandwidth, and unreliable. In football, set-plays would be a much more prevalent part of the locker-room agreement.

As mentioned in Section 3.8, there are several examples of non-sports-related PTS domains, including multi-spacecraft missions [Stone 97], search and rescue domains, hospital/factory maintenance [Decker 96b], battlefield combat [Tambe 97], and network routing. Another PTS domain with single-channel, low-bandwidth, unreliable communication is multi-robot systems with audio communication [Fujita and Kageyama 97]. Applying the team member agent architecture to some of these domains is an important direction for future work.

Layered Learning

More learned layers. The layered learning implementation in this dissertation consists of three learned layers ranging from an individual behavior to a team behavior. Section 4.2.2 details the possible extension of this implementation to include another team behavior and an adversarial behavior. Successfully implementing these or other additional learned layers would be an interesting avenue for future research.

Automatic task decomposition. Layered learning works with a given task decomposition. However, it could be combined with a method for learning task decompositions. As presented in Section 4.1,

> let A be an algorithm for learning task decompositions within a domain. Suppose that A does not have an objective metric for comparing different decompositions. Applying layered learning on the task decomposition and quantifying the resulting performance can be used as a measure of the utility of A's output.

The creation of an algorithm A for learning task decompositions is a challenging open research direction.

Error propagation. In this dissertation, each of the learned layers is validated in a controlled testing scenario, demonstrating the effectiveness of the learned behavior. However, no study is made of the propagation of errors from one layer to the next. It is possible that errors at initial layers could hurt performance at all subsequent layers. However, since learned layers are trained individually, it is also possible that the learning at subsequent layers could compensate for earlier errors and thus render them inconsequential. A detailed study of error propagation and compensation within layered learning implementations is a promising area for future research.

Layered learning is applied in simulated robotic soccer in this dissertation. A possible next step is to apply layered learning in real robotic soccer as well. In this case, the same task decomposition as is used in simulated robotic soccer might apply.

In general, layered learning applies in domains that are too complex for tractably learning a mapping directly from an agent's sensory inputs to its actuator outputs. Many real-world domains fit this characteristic. For example, layered learning could be applied in a multi-agent search and rescue domain. Like in robotic soccer, agents could learn individual, multi-agent, team, and adversarial behaviors.

TPOT-RL

Choosing a representation. One of the key responsible factors for TPOT-RL's success in both simulated robotic soccer and network routing is the choice of input representation for learning. In particular, the action-dependent feature function $e(s, a)$ determines the representation of the learned function. In the case of robotic soccer, the input representation is constructed from the previously learned layer in layered learning.

In both implementations of TPOT-RL in this dissertation, a single input representation is successfully used. However, other representations are possible. For example, in robotic soccer, rather than using the learned pass-evaluation function as e, the agents could perhaps learn based on the distance of the closest opponent to the pass target.

Quantifying the effect of learned TPOT-RL behaviors as a function of the input representation used would be an interesting result.

Choosing a reward function. Similarly, the implementations of TPOT-RL in this dissertation vary the agents' reward function only minimally. As with all reinforcement learning methods, the reward function has an important effect on what is learned. It would be interesting to quantify the effects of the reward function on the learned behaviors when using TPOT-RL.

As presented in Section 7.5, TPOT-RL applies in domains with the following characteristics:

- There are multiple agents organized in a team.
- There are opaque state transitions.
- There are too many states and/or not enough training examples for traditional RL techniques to work.
- The target concept is non-stationary.
- There is long-range reward available.
- There are action-dependent features available.

In addition to simulated robotic soccer and network packet-routing, such domains include information networks, distributed logistics, and rescue missions. As with all of the thesis contributions, it is a valuable direction of future research to apply TPOT-RL to these other multi-agent domains.

10.3 Concluding Remarks

In short, this dissertation contributes techniques for building agents in a particularly complex class of real-world domains: real-time, noisy, collaborative and adversarial multi-agent environments. I believe that AI agents will be applied to this type of domain more and more frequently in the near future. I hope that the thesis contributions will prove useful in addressing the problems that arise in these domains, and, ultimately, will improve our understanding of what it takes to build complete, AI agents.