Adaptive Teamwork Coordination using Graph Matching over Hierarchical Intentional Structures

Susannah Soon[1,2], Adrian Pearce[1], and Max Noble[2]

[1]*Department of Computer Science and Software Engineering, The University of Melbourne, Victoria, 3010, Australia*

[2]*ADI Limited, 20-22 Stirling Highway, Nedlands, WA, 6009, Australia*

{*susannah, pearce*}*@cs.mu.oz.au, max.noble@adi-limited.com*

Abstract

Many existing teamwork coordination approaches recognise team intention by using communications, and/or by identifying plan execution through observing agent actions. However, problems may arise when such information is unavailable, or when agents are not observable at runtime. This paper presents a new agent coordination strategy, called Rolegraphs, that represents and recognises team intentions without requiring full knowledge of plans, or complete observations. The strategy relies on the role relationships formed within hierarchical teamwork structures. A graph matching approach is used to interpret these hierarchical structures, and to recognise team intentions at runtime. The main contribution of this work is the use of efficient graph matching techniques to dynamically coordinate diverse and large-scale teams, where communications, observations, or plan details are incomplete. The Rolegraph coordination strategy is shown to be compatible with existing coordination approaches in the literature, and promises to improve the flexibility and robustness of coordination.

1. Introduction

Arranging agent relationships, or roles, hierarchically presents several advantages for coordination purposes. First, coordination effort is reduced as agent and team responsibilities, beliefs, actions, and intentions are more constrained, and described by their organisational relationships [4, 20, 21]. Second, important team relationships are captured while abstracting away unnecessary low-level details.

However, role-based hierarchies present some disadvantages for agent coordination, including the complexity of structural analysis, and maintenance. This particularly concerns heterogeneous or large-scale structures, whose teams vary in size and configuration, and are developed using different agent languages, or different teamwork or reasoning models [16]. Interpretation of these structures is made difficult as roles are represented at different scales, and have alternative meanings within particular contexts.

Current multiagent teamwork coordination strategies rely on theoretical teamwork frameworks such as [3, 7, 11], and typically involve the recognition of agent mental states, or Belief-Desire-Intention (BDI) [16] states. They generally use one or more sources of information, including communications [20], observations [8], and plan details [8-10, 17]. However, these coordination approaches are often unreliable when information is incomplete. Communication-based approaches are limited when communications are unreliable, conflicting, or costly [8, 9, 20], for example, when plan failure is uncommunicated [20]. On the other hand, plan recognition [8, 17, 20] involves observing agent activities to identify the steps of current plan execution. This approach suffers when either the planning steps are unknown ahead of time, or when observations are unavailable, for example, when agents are not observable.

This paper builds on the authors' prior work [18], which presents coordination of agent services in military desktop decision support systems, using hierarchical teamwork structures. The present work describes the Rolegraph coordination strategy (RCS) for agent teamwork, which can operate with only partial information available to each agent at runtime. This is achieved using graph matching principles to interpret hierarchical role relationships that represent team intentions. Incomplete plan information is overcome using hierarchical structure abstraction, and partial observations are supplemented by reasoning over the situational environment, and forming assumptions according to the hierarchical structure.

AAMAS'04, July 19-23, 2004, New York, New York, USA.

We use a polynomial time graph matching algorithm [13] to perform efficient team structure comparisons. This allows us to quickly identify structures, and improves the scalability of the approach for large hierarchies. However, graph matching provides fewer advantages when non-hierarchical structures are considered, as less implicit relational information is captured. Hierarchies allow relationships to be abstracted and simplified, enabling graph techniques to be practically applied over these structures. We therefore capitalise on the synergy between the advantages associated with graph matching, and the advantages of hierarchical role relationships, by using them in combination.

Several existing teamwork approaches express role relationships using team hierarchies [20, 21]. Unlike these approaches, our Rolegraph coordination strategy (RCS) [19] uses hierarchical role relationships situated in environmental contexts, as explicit representations of team intention. These hierarchies are matched against templates of known behaviour to identify a team's intention to perform a particular task, within a given context. Using the hierarchical graph matching approach reduces the BDI reasoning required to recognise team intentions.

The implications of this research include the ability to perform interleaved and adaptive coordination for heterogeneous and large-scale teams. We consider heterogeneous structures as those differing in size and configuration. Such configurations are unintelligible to other teams, as they represent command structures (e.g. centralised, or distributed) differently. A practical example of heterogeneous team structures are two opponent teams developed using separate teamwork models. For example, a large-scale team structure may include additional representation of intermediate command. In addition, it is difficult to associate observations with particular team plans that concern different scale teams and tasks. Therefore, as heterogeneous team structures scale-up, plan and observation information is less useful to infer team intention.

The RCS addresses these heterogeneity and scalability issues by simplifying hierarchies using an abstraction process (section 3.4), removing less relevant low-level detail so that structures are easier to interpret. For example, coordinating high level teams instead of their subteams. In addition, by matching over whole or significant partial hierarchies, the RCS effectively considers entire team relationships at an abstract level.

These characteristics enable the RCS to dynamically recognise the intentions of various size heterogeneous team structures. This allows the RCS to perform adaptive, and interleaved coordination, required in large-scale heterogeneous team structures where various levels of coordination are necessary due to different heterogeneous team interactions.

AAMAS'04, July 19-23, 2004, New York, New York, USA.

For ease of exposition, we demonstrate this work in the pursuit domain [2], a distributed artificial intelligence problem often used to examine multiagent system coordination. We also comment on the complexity of the approach.

2. Problem Definition

Our pursuit world considers a 10 × 10 grid, containing 2 teams of 4 predators, and 2 prey, positioned within grid squares. Predator teams can pursue either prey. Prey are captured when 4 predators of a single team surround it in diagonally adjacent squares. Predator and prey move alternatively in sequence, and cannot move beyond the grid boundary. Predators can partially communicate with, and observe other predators. Prey cannot observe predators' positions or currently executing tasks. Work duplication is reduced when predators pursue different prey. Therefore, successful coordination results when predators recognise which prey the other predator team is attempting to capture.

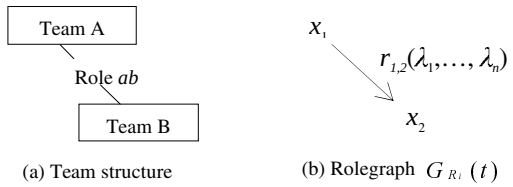
Terminology: A *team* is an agent reasoning entity that collaborates with other teams to achieve common goals. Teams require roles according to the organisational hierarchy role they perform. *Roles* specify the relationship, including behaviour and responsibilities between teams and their subteams. Roles are filled by subteams, or individual agents that have the capability to perform the role. Roles are unallocated when no team satisfies the role's required capabilities. *Teamwork structures* specify hierarchical team-role-team relations (see figure 1). Teams are initially allocated to roles. At runtime, environmental changes cause *dynamic team formation* to occur. Subteams change in applicability, and are reallocated to suitable roles.

Rolegraph Definition: Hierarchical teamwork structures dynamically model organisational hierarchy role relationships between teams with respect to the changing environment. Rolegraphs are extracted from dynamic teamwork structures at runtime. Rolegraph examples include predator team roles, or roles representing tasks performed to assist coordination of predator teams. The term Rolegraph is independent of Role Graph [14] used in computer security.

A Rolegraph is described using a tertiary relationship $G_R(t) = (x_1, x_2, r_{1,2}(\lambda_1,\ldots,\lambda_n))$. It contains unique identifier unary labelled vertices x_i, and x_j, representing role-tendering, and role-filling teams. Rolegraphs also contain labelled and attributed edges $r_{i,j}(\lambda_1,\ldots,\lambda_n)$ representing binary role relationships between tendering and filling teams.

Definition 1. A ***Rolegraph*** is a directed acyclic graph, specifically, an *N-ary* tree. It is defined by the tuple $G_R(t)$

$= <V_{Ri}, V_{Rj}, E_R>$, with vertices $V_{Ri} = x_i$, $V_{Rj} = x_j$, and edges $E_R = r_{i,j}(\lambda_1,\ldots,\lambda_n)$.



(a) Team structure (b) Rolegraph $G_{Ri}(t)$

Figure 1. a) a team structure; b) a corresponding Rolegraph.

Figure 1 shows equivalent team structure and Rolegraph parts. Figure 2 shows Predator team 2's Rolegraph configuration at time t=0, requiring four Predator roles. These roles are filled by Attack, Surround, or Distract teams according to Predator team 2's current intention.

Predator team intentions are expressed using hierarchy levels *a*, *b* and *c*. Figure 2 shows only the level *a* configuration describing possible Predator team intentions of attacking, surrounding, or distracting the prey. Level *b* expresses the types of attack, surround or distract performed. Level *c* describes the task decomposition required to conduct the attack, surround, or distract tasks.

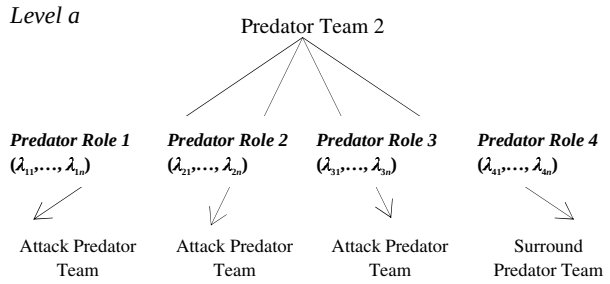


Figure 2. Level *a* Predator Team 2 Rolegraph.

A binary role relationship $r_{i,j}(\lambda_1,\ldots,\lambda_n)$ specifies the unique relationship and responsibilities between teams and their subteams, where $r_{i,j}$ is the unique identifier for the role. The role relationships consist of multiple attributes, $\lambda_1,\ldots,\lambda_n$. Role relationships closely map team relationships within corresponding team structures, and apply directionally between teams, as indicated by the Rolegraph's directed edge.

Definition 2. ***Attributes*** are values defining a role relationship. They can be either single values, or sets of values.

A role relationship contains five attributes, $\lambda_1,\ldots,\lambda_5$. λ_1 is the role name. λ_2 are capabilities, the actions that a subteam must be able to perform in order to fill the role. λ_3 are role specific beliefs. These are beliefs that the filling team must possess in order to fill the role. λ_4 are the relationships and norms between teams within the teamwork structure organisational hierarchy. They are expressed using individual and joint intentions, beliefs, and plans; and active roles and attached teams. λ_5 are teams suitable to attach to this role. Table 1 shows role attributes, and values for particular Predator team roles.

Table 1. Predator role attributes.

	Predator Role 1, 2, 3	**Predator Role 4**
λ_1		
λ_2	Surround, or attack prey. Visual range: 6 squares. Speed: fast. Movement range: 2 squares. Movement direction: north, south, east, and west. Intelligence: high.	Surround, or distract the prey. Cannot attack prey, Visual range: 4 squares. Speed: slow. Movement range: 1 square Movement direction: east and west. Intelligence: average.
λ_3	Prey is not captured. Prey position is (x, y).	Prey is not captured. Prey position is $(x \pm 1, y \pm 1)$.
λ_4	Able to command other Predator teams.	Able to command Predator teams with lesser capability.
λ_5	Surround or attack teams.	Surround or distract teams.

Example 1: In figure 2, Attack, Attack, Attack, and Surround teams attach to Predator roles 1 to 4 respectively. Referring to λ_2, and λ_5 in table 1, and environmental conditions (not described), this configuration is inferred to represent a surround or attack team intention. This inference must be further verified using graph matching to compare the Rolegraph against Surround and Attack templates. If more detailed information concerning Predator team 2's activities is required, level *b* and *c* Rolegraphs must be matched against templates.

Template Definition: Templates represent particular team activities by specifying configurations including teams, subteams, and role relationship attribute values. Templates can be mathematically and schematically represented similar to Rolegraphs (figure 1b). A template is described using a tertiary relationship $G_{Ti} = (y_1, y_2, r_{1,2}(\lambda_1,\ldots,\lambda_n))$. It contains unique identifier unary labelled vertices y_i, and y_j, representing role-tendering, and role-filling teams. Templates also contain labelled and attributed edges $r_{i,j}(\lambda_1,\ldots,\lambda_n)$ representing the binary role relationship between tendering and filling teams.

Definition 3. A ***template*** is a directed acyclic graph, specifically, an *N-ary* tree. It is defined by the tuple $G_T = <V_{Ti}, V_{Tj}, E_T>$, with vertices $V_{Ti} = y_i$, $V_{Tj} = y_j$, and edges $E_T = r_{i,j}(\lambda_1,\ldots,\lambda_n)$.

Templates are predefined, and can be generic, or domain specific. Generic templates represent information such as organisational command. Domain specific template examples include those in the pursuit domain, such as Predator team Attack, Surround, and Distract templates. Knowledge acquisition for pursuit domain

AAMAS'04, July 19-23, 2004, New York, New York, USA.

templates is retrieved from feasible predator and prey activities.

3. Rolegraph Coordination Strategy

The Rolegraph coordination strategy provides several characteristics to assist intention recognition.

3.1 Reasoning Capability

Each team in a Rolegraph hierarchy possesses a *reasoning capability* that is used to infer team intentions. The *reasoning capability* overcomes unavailable observation and plan information by forming hypotheses of team mental states. Hypotheses are formed by reasoning over the team's relationship to other teams, with respect to current environment conditions. At runtime, the *reasoning capability* identifies relevant Rolegraphs to test the hypotheses. The *reasoning capability* can examine its own, and other team Rolegraphs from various scales, and perspectives to test hypotheses such as whether teams have the potential to coordinate, whether a team possesses information of interest, or whether a team is an opponent.

Hypothesis formation relies on knowledge of a team's likely intentions. For example, it must first be recognised that predator teams in the pursuit domain aim to capture particular prey. The following hypothesis is formed h_1: "*Predator team 2 intends to surround Prey 1*".

The *reasoning capability* identifies the intentions of a team of interest, and establishes if they are consistent with intentions required for coordination according to theoretical teamwork framework principles [3, 7]. These are represented in the Rolegraph coordination strategy (RCS) using beliefs, plans, and context conditions.

Preliminary Rolegraph interpretation is obtained by examining available roles, and team role filling. The *reasoning capability* then performs more intensive interpretation by matching Rolegraph configurations against templates. In the pursuit domain, Predator team 1's *reasoning capability* tests h_1 by interpreting Predator team 1 and 2's internal and external Rolegraph relationships.

3.2 Hierarchical Intentional Structures

RCS uses hierarchical structures to assist intention recognition reasoning, as a team's *reasoning capability* can introspect and analyse other team activities from various perspectives (*inter*- or *intra*-hierarchy). It enables scalable coordination, as teams can potentially coordinate with, or recognise, the activities of any other teams in the hierarchical structure. Coordination is flexible as teams can evaluate coordination potential from their individual perspectives. Coordination is also robust as each team can perform coordination reasoning, should other team members fail, or become unavailable.

The Rolegraph hierarchy represents an intentional structure as valid relationships must be maintained between teams and their subteams. Team formation is restricted by the ability of the subteam to perform the roles required by the higher level team whilst maintaining consistency with other relationships in the hierarchy. For example, relationships existing lower in the hierarchy enforce relationships that must exist higher in the hierarchy, for the hierarchy to be consistent and for particular activities to be performed.

3.3 Retrieving Adaptive Rolegraphs

Teams require up-to-date views of relevant Rolegraph configurations to conduct intention recognition. Although configurations can be updated using explicit communications, unreliable communications and overheads exist. Instead, individual teams hold beliefs of approximate intentional Rolegraph configurations.

Access to dynamic Rolegraph configurations is straightforward when team structures are defined and recognised in the same system. In such cases, teams have full observation of other team actions, and a team's belief of Rolegraph configurations is simply updated at runtime.

Rolegraph configurations are harder to obtain when team structures are developed using different systems (e.g. opponent teams). The team must rely on possibly unavailable observations of agent action. An abstract and reliable way to form team structure configuration beliefs is to consider peripheral domain specific environmental information. These are called *features*, and include situational observations, events, and facts. Beliefs of team structure configurations evolve as reasoning is performed over such information together with existing beliefs of capabilities, and historical information.

Definition 4. ***Features*** are environmental observations, facts or events, describing how the agent or team is situated in its environment.

Example 2: In the pursuit domain, *features* include the believed adjacency of Predator teams, based on estimates of a predator's possible speed depending on whether it is attacking, surrounding, or distracting; and physical environmental characteristics such as danger zones. Suppose Predator team 1's *reasoning capability* believes the adjacency of Predator team 2's team-mates to each other, and to Prey 1, is less than 2 grid squares. Other environmental *features* also suggest that the area populated by predators and prey, is bordered by grid boundaries on two sides, and evolving danger zones on the remaining sides. As this limits predator movement, Predator team 2 is assumed to be *surrounding* Prey 1.

AAMAS'04, July 19-23, 2004, New York, New York, USA.

As beliefs are derived from observed *features*, they provide abstract, long-term, and less accurate representations of team activities, compared to observing direct team activities. However, *features* contribute to the Rolegraph's intentional nature by representing the team's overall situation.

3.4 Abstracting Rolegraph Structures

Prior to matching, Rolegraph configurations are abstracted to increase Rolegraph and template compatibility. Abstraction is immediately evident in RCS, as an assumption is that Rolegraph runtime information is incomplete, leading to approximate intention recognition. In addition, using approximate Rolegraph information is feasible in dynamic environments, as precise configurations identified in one time instance may be inapplicable in another. The *reasoning capability* performs further runtime abstraction (definition 5), to retrieve useful information for intention recognition.

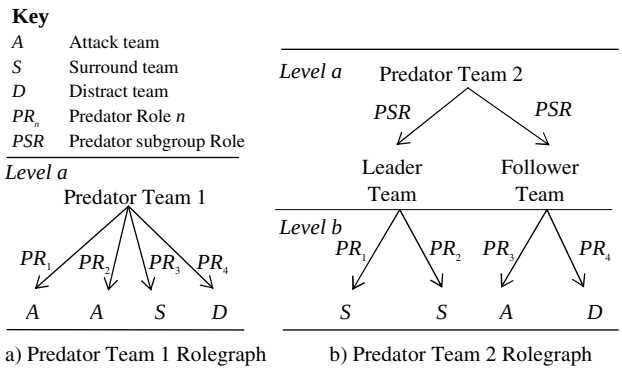
Definition 5. ***Abstraction*** simplifies Rolegraphs by considering relevant Rolegraph subgraphs or levels of detail.

A suitable abstraction level simplifies Rolegraph configurations, and extracts only specific details for consideration. This reduces the reasoning required to perform Rolegraph matching and interpretation. Fewer Rolegraph matching iterations are required to identify agent mental states, reducing the number of hypotheses to be tested. For instance, example 1 level *a*, examines a team's high level configuration.

The *reasoning capability* abstraction selection process involves identifying team structure hierarchies of interest to test hypotheses. The *reasoning capability* uses *selectTeamStructure*() to select the most useful team structure hierarchy T_x, where $T_x \in T$ and $T = \{T_1, T_2, \ldots, T_n\}$ is the set of team structure hierarchies of interest. *navigateLevels*() selects a useful level of abstraction, l_x in the given hierarchy. The abstraction level, l_x represents the section of the hierarchy of interest, ranging from a row, to particular branches, or particular levels of detail contained in the hierarchy. *informationSufficiency*(l_x) determines if the selected teamwork hierarchy abstraction level, l_x, provides sufficient information for coordination (details omitted).

Rolegraphs are divided into levels (figure 3) representing particular activities or organisational command. Choosing the abstraction level involves selecting particular team structure level (s). *navigateLevels()* is used to navigate up or down the levels to increase or decrease abstraction, according to information sufficiency.

Example 3: Figure 3 shows Predator teams 1 and 2 constructed using heterogeneous configurations. Predator team 1 is represented using a single level, *a*, known as configuration type 1. Predator team 2 is represented over levels *a* and *b*, known as configuration type 2, which includes an intermediate level of command describing Predator team subgroups. From a high-level perspective, Predator team 1's single level is equivalent to Predator team 2's two levels. To test h_1, the Predator team 1 requires knowledge of Predator team 2's intermediate level strategies, and refers to level *a* specifying whether the Predator subgroups perform *leader or follower* activities. Predator team 1 also requires detailed knowledge of the Predator team 2's current activities, and navigates to level *b* to retrieve this information.



a) Predator Team 1 Rolegraph b) Predator Team 2 Rolegraph

Figure 3. Heterogeneous Rolegraphs: a) Predator team 1; b) Predator team 2.

3.5 Rolegraph Matching

The *reasoning capability* matches relevant Rolegraphs against templates to test intention hypotheses. Matching compares Rolegraph structural configurations. This involves comparing attributed edges (role relationship attributes), and graph vertices (teams) to identify templates that best match the Rolegraph being considered.

Although the subgraph isomorphism problem is known to be NP complete [1], the exact graph isomorphism problem is not known to be in NP, and is suspected to be neither in NP or P. For Rolegraph matching, we use a polynomial time graph matching algorithm, the Kuhn-Munkres algorithm [13], that results in a weaker form of isomorphism, known as a bipartite matching. Graph isomorphism requires exact 1-1 matching of edges in graph G_1 to edges in graph G_2. In contrast, exact bipartite matching does not require a 1-1 matching between edges.

A bipartite graph $G_b = (V_b, E_b)$ contains the set of vertices V_b, which can be divided into two partites V_1 and V_2 such that no edge E_b, connects vertices in the same set. We assign Rolegraph vertices to partite V_1, and template

AAMAS'04, July 19-23, 2004, New York, New York, USA.

vertices to partite V_2. The weightings of the edges between V_1 and V_2 are an approximation of the degree the vertices correspond to one another. The matching algorithm is then used to find the maximal weighted matching of the bipartite graph. That is, the largest weighted set of edges from E_b such that each vertex in V_b is incident to at most one edge.

The *weighting* w, between the vertices of Rolegraph $G_{R_i}(t)$ and template G_{T_i} is determined by the summation of attribute comparisons between corresponding roles, $w = \forall_{\lambda_k \in G_{R_i}(t), \lambda_k \in G_{T_i}} \sum f_k(\lambda_k, \lambda_k)$. The attribute comparisons are conducted using an attribute specific *comparison function*, $f_k(\lambda_k, \lambda_k)$, which can be equivalence, or a generalisation testing whether attributes belong to sets or ranges of numeric, symbolic, or syntactic values. The maximum weighted graph identifies the best matched template. Bipartite Rolegraph matching is empirically demonstrated in [19]. Figure 4 shows Rolegraph versus Template matching.

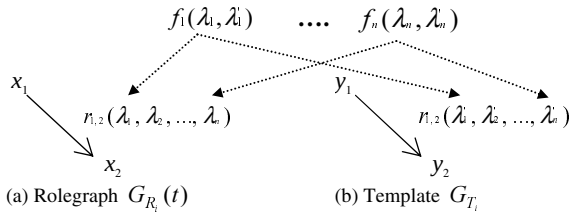


Figure 4. Rolegraph vs. template matching.

During matching, team *reasoning capabilities* define the *comparison function* by indicating the required level of attribute generalisation. Generalisation involves specifying allowable attribute values. Attribute generalisation involves identifying role attributes of interest and levels of detail at which they are analysed. Generalisation affects the degree to which Rolegraph role relationships need to match those of the template to be considered equivalent. For example, if a template contains highly generalised role relationships then several templates are suitable to match the Rolegraph. Fewer Rolegraphs are suitable to match the template if its role relationship attributes are specific. In different situations, coordination may require generalised or specific matching, or that the template need only match certain parts of the structure, at varying levels of generality.

Bipartite matching also matches template and Rolegraph roles that are not correspondingly ordered, if their role attributes are equivalent. For example, when a Rolegraph has two similar roles filled by similar teams, a match can be made between Rolegraph and template roles that are in reverse order.

Comparing templates to identify suitable coordination decisions is equivalent to testing context conditions of coordination rules [20] before invoking plan execution. However, coordination rules present disadvantages as they are possibly domain specific. In addition, coordination rule invocation conditions are inflexible as they cannot be approximately matched, or concisely represent information. In contrast, templates can be modified easily to match whole or partial team structures. Templates can compare relevant detailed information, such as situational, and intentional information, using attribute generalisation. Templates can also be easily stored, and interpreted using graph matching.

During matching the *reasoning capability* uses its beliefs of events that should occur to search for patterns to trigger recognition, and then reasons about the observation [8] to infer intention. It uses matching results to infer mental models of agent intentions relative to their goals in the environment, and according to teamwork framework principles [3, 7]. Once team intentions are ascertained, suitable coordination actions, represented as plans, or generic coordination rules, are selected. For example, requesting teams to share information or tasks.

4. Scenario

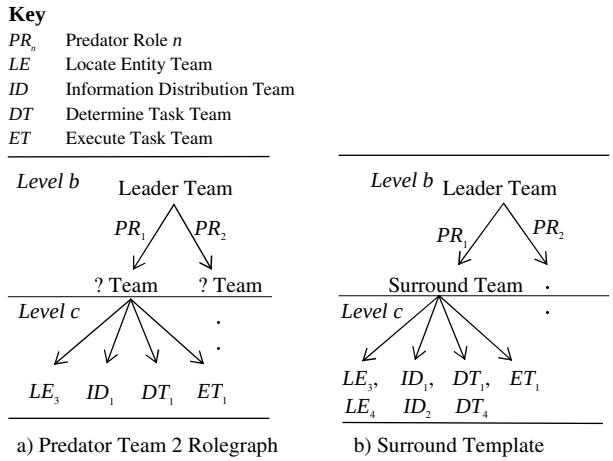
We demonstrate how the RCS performs intention recognition, by showing how Predator team 1's reasoning capability tests h_1 using multi-level Rolegraph matching. Consider an extended Predator team 2 Rolegraph where each level *b* team has the following level *c* roles: LocateEntityRole (*LE*), InformationDistributionRole (*ID*), DetermineTaskRole (*DT*), and ExecuteTaskRole (*ET*). Each role has corresponding teams that may be applicable to attach. For example, the role *LE* has available teams $\{LE_1,\ldots,LE_n\}$. Testing h_1 involves determining which of the Attack, Surround, or Distract teams fills Predator Role 1. Relevant templates for bipartite matching of higher level relationships are constrained by knowledge of lower level relationships. Previous hypothesis testing confirms lower level (level *c*) hierarchy configurations. In particular, the team attaching to Predator Role 1 has the configuration LE_3, ID_1, DT_1, ET_1. In order to determine if the Attack, Surround, or Distract teams are applicable, the *reasoning capability* considers the subteams that can be exhibited by each of these teams:

Attack: $\{LE_2, LE_3\}, \{ID_1, ID_3\}, \{DT_1, DT_3\}, \{ET_1\}$
Surround: $\{LE_3, LE_4\}, \{ID_1, ID_2\}, \{DT_1, DT_2\}, \{ET_1\}$
Distract: $\{LE_3, LE_4\}, \{ID_2\}, \{DT_1, DT_2\}, \{ET_1, ET_2\}$

The Attack and Surround teams contain subteams consistent with the observed level *c* configuration. Therefore, level *b* Attack and Surround templates are suitable templates to determine the level *b* configuration. Figure 5 compares Predator Team 2's Rolegraph with the

AAMAS'04, July 19-23, 2004, New York, New York, USA.

Surround Template. Bipartite matching of Rolegraph and template PR_1 and PR_2, identifies the Surround template as the best match, the Surround team is considered to fill PR_1.



a) Predator Team 2 Rolegraph b) Surround Template

Figure 5. Predator Team 2 Rolegraph vs. Surround template.

Execution Sequence Analysis: Evaluation of the RCS involves execution sequence analysis. Execution sequences evolve according to accurate or inaccurate Rolegraph recognition as scenario hypotheses are tested. Consider the pursuit world scenario (where Predator team *n* is denoted P_n):

1. P_1 recognises P_2's intention to surround Prey 1.
2. P_2 recognises P_1's intention to surround Prey 2.
3. P_1 and P_2 surround and capture Prey 1 and 2.

Steps 1 and 2 correspond to hypotheses h_1 and h_2. When Rolegraph recognition is accurate and all scenario hypotheses are true, the resulting sequence is: 1→2→3→end. Alternative execution sequences result when recognition is inaccurate and the hypotheses are false. Two examples follow:

1. P_1 fails to recognise P_2's intention to surround Prey 1.
2a. P_1 intends to surround Prey 1.
3a. P_1 and P_2 surround Prey 1.

The resulting execution sequence is: 1→2a→3a→end.

1. P_1 recognises P_2 intention to surround Prey 1.
2. P_2 fails to recognise P_1's intention to surround Prey 2.
3b. To coordinate capture activities, P_2 waits until it believes P_1 forms an intention to surround Prey 2.

The resulting execution sequence is: 1→2→3b→end.

5. Complexity of Rolegraph Matching

A Rolegraph is a subgraph of a directed acyclic graph, specifically, a branch of a directed *N-ary* tree. Time complexity of performing operations and managing these kind of structures is consistent with balanced N-ary trees, that is, $O(n\log n)$, where n represents vertices (or equivalently edges). We implement the Kuhn-Munkres algorithm [13] for maximum weighted bipartite matching, having worst case complexity $O(n^4)$, where n represents the number of vertices (or equivalently edges). Gabow [5] presents an improved algorithm with complexity $O(n^3)$.

A single pass through the matching algorithm is required for each Rolegraph versus template comparison. A comparison is performed for each template, where the number of templates is denoted p. Therefore, the worst case complexity of the Rolegraph matching algorithm is $O(p(n^3))$.

For larger numbers of agents (over 100), linear time bipartite matching algorithms based on disjoint set unions are available. For example, an algorithm running in $O(nm\log(n^2/m))$ time on an n-vertex, m-edge graph based on a dynamic tree data structure, that admits efficient distributed and parallel implementations (see Goldberg and Tarjan [6] for details).

6. Related Work

The STEAM general purpose teamwork model [20], relaxes the full plan knowledge requirement by incorporating aspects of SharedPlans theory [7] within its reactive plan hierarchy. SharedPlans theory states that Partial Shared Plans (PSPs) are sufficient for coordination. Therefore, team members need only be aware of other teams' intentions to execute a plan, rather than requiring knowledge of full plan details. This concurs with the RCS, which emphasises intention knowledge over plan detail. STEAM implementations [10, 15] also use organisational hierarchies, selective communication, observation of action, role-monitoring, and replanning on failure to infer intentions. The advantage of RCS is that it recognises commitment to intention by matching intentional teamwork hierarchies, rather than depending on possibly unreliable communications, and observations.

A conceptual model for plan-based BDI mental state team modelling [17] infers agent mental states using partial knowledge of observed action, hypothesis testing of possible teams and structure, and abstraction of low-level details when plan knowledge is incomplete. A limitation is the numerous team mental state and structure hypotheses stored and reasoned over, leading to possible overheads and inaccuracies. RCS uses abstraction and graph matching over intentional hierarchies to reduce required hypotheses.

Roles and graphs have been combined for coordination [12]. The RCS differs as it performs coordination by using

AAMAS'04, July 19-23, 2004, New York, New York, USA.

graph matching to interpret situational and intentional role relationship hierarchies.

7. Conclusion and Future Work

This paper contributes a dynamic teamwork coordination approach that overcomes incomplete knowledge of communications, plans, and observations. The RCS incorporates graph matching of intentional role based hierarchies, into BDI reasoning. The approach provides flexibility and robustness, allowing heterogeneous and large-scale teams with limited information to be coordinated. In addition, the generality of the approach is demonstrated by the modularity of the *reasoning capabilities*, and graph matching components that can be substituted with different agent reasoning models, or matching processes.

We are presently conducting further empirical evaluation, beyond that reported in [19]. This work investigates RCS's ability to coordinate large-scale and heterogeneous structures, and involves the random generation and analysis of execution sequences over time.

8. Acknowledgements

This research is supported by an Australian Research Council Linkage grant, and ADI Limited.

9. References

[1] H. G. Barrow and R. M. Burstall, "Subgraph Isomorphism, Matching Relational Structures and Maximal Cliques", *Information Processing Letters*, **4**(4): pp. 83-4, 1976.

[2] M. Benda, V. Jagannathan, and R. Dodhiawala, "On Optimal Cooperation of Knowledge Sources - an Experimental Investigation," Boeing Advanced Technology Center, Boeing Computing Services, Seattle, Washington, Technical Report BCS-G2010-28, July 1986.

[3] P. Cohen and H. Levesque, "Teamwork", *Nous, Special Issue on Cognitive Science and AI*, **25**(4): pp. 487-512, 1991.

[4] E. H. Durfee, "Practically Coordinating", *AI Magazine*, **20**(1): pp. 99-116, 1999.

[5] H. N. Gabow, "An Efficient Implementation of Edmonds' Algorithm for Maximum Matching on Graphs", *Jour. ACM*, **23**: pp. 221-234, 1976.

[6] A. V. Goldberg and R. E. Tarjan, "A New Approach to the Maximum-Flow Problem", *Journal of the Association for Computing Machinery*, **35**(4): pp. 921-40, 1988.

[7] B. J. Grosz and S. Kraus, "Collaborative Plans for Complex Group Action", *Artificial Intelligence*, **86**(2): pp. 269-357, 1996.

[8] C. Heinze, S. Goss, and A. Pearce, "Plan Recognition in Military Simulation: Incorporating Machine Learning with Intelligent Agents", in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI99) Agent Workshop on Team Behaviour and Plan Recognition, Stockholm, pp. 53-63, 1999.

[9] M. Huber and E. Durfee, "On Acting Together: Without Communication", in Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI'96), 1996.

[10] G. A. Kaminka and M. Tambe, "Robust Agent Teams Via Socially-Attentive Monitoring", *Journal of Artificial Intelligence Research*, **12**: pp. 105-47, 2000.

[11] D. Kinny, M. Ljungberg, A. Rao, E. Sonenberg, G. Tidhar, and E. Werner, "Planned Team Activity", in Artificial Social Systems. 4th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW '92. Selected Papers, S. Martino al Cimino, Italy, pp. 227-56, 1994.

[12] J. Kok, M. Spaan, and N. Vlassis, "An Approach to Noncommunicative Multiagent Coordination in Continuous Domains", in Proceedings of Benelearn'02 Annual Machine Learning Conference of Belgium and The Netherlands, Utrecht, The Netherlands, pp. 46-52, 2002.

[13] H. W. Kuhn, "The Hungarian Method for the Assignment Problem", *Naval Res. Logist. Quart*, **2**: pp. 83-97, 1955.

[14] M. Nyanchama and S. Osborn, "The Role Graph Model and Conflict of Interest", *ACM Transactions on Information and Systems Security*, **2**(1): pp. 3-33, 1999.

[15] D. Pynadath and M. Tambe, "An Automated Teamwork Infrastructure for Heterogeneous Software Agents and Humans", *Autonomous Agents and Multi-Agent Systems*, **7**: pp. 71-100, 2003.

[16] A. Rao and M. Georgeff, "Modeling Rational Agents within a BDI-Architecture", in Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, pp. 473-484, 1991.

[17] E. Sonenberg and G. Tidhar, "Observations on Team-Oriented Mental State Recognition", in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI99) Agent Workshop on Team Behaviour and Plan Recognition, Stockholm, 1999.

[18] S. Soon, A. Pearce, and M. Noble, "Modelling the Collaborative Mission Planning Process Using Dynamic Teamwork Structures", in Proceedings of the 2nd International Joint Conference on: Autonomous Agents and Multiagent Systems, Jul 14-18 2003, Melbourne, Australia, pp. 1124-1125, 2003.

[19] S. Soon, A. Pearce, and M. Noble, "A Teamwork Coordination Strategy Using Hierarchical Role Relationship Matching", in Proceedings of the 1st International Workshop on Computational Autonomy - Potential, Risks, Solutions (AUTONOMY 2003), held in conjunction with AAMAS 03, Melbourne, Australia, 2003.

[20] M. Tambe, "Towards Flexible Teamwork", *Journal of Artificial Intelligence Research*, **7**: pp. 83-124, 1997.

[21] G. Tidhar, A. S. Rao, and E. A. Sonenberg, "Guided Team Selection", in Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS-96), Kyoto, Japan, pp. 369-76, 1996.

AAMAS'04, July 19-23, 2004, New York, New York, USA.