A Modular Framework for Decentralised Multi-Agent Planning

(Extended Abstract)

Rafael C. Cardoso, and Rafael H. Bordini
Faculdade de Informática, PUCRS
Porto Alegre – RS, Brazil
rafael.caue@acad.pucrs.br,rafael.bordini@pucrs.br

ABSTRACT
Multi-agent systems often require runtime planning, which remains an open problem due to the existing gap between planning and execution in practice. Extensive research has been carried out in centralised planning for single-agent systems, but so far decentralised multi-agent planning has not been fully explored. In this paper, we extend existing multi-agent platforms to enable decentralised planning at runtime. In particular, we put forward a planning and execution framework called Decentralised Online Multi-Agent Planning (DOMAP). Experiments with a planning domain we developed on flooding disaster scenarios show that DOMAP outperforms 4 other state-of-the-art multi-agent planners, particularly in the most difficult problems.

Keywords
multi-agent planning; multi-agent systems; distributed problem solving; online planning

1. INTRODUCTION
Research on automated planning has been mostly focused on centralised planning. Although it is possible to adapt single-agent techniques to work in a decentralised way, as in [6], distributed computation is not the only advantage of using Multi-Agent Planning (MAP). By allowing agents to do their own individual planning, the search space is effectively pruned, which can potentially decrease planning time on domains that are naturally distributed, and allow agents to keep some privacy within the system. MAP has received increasing attention recently [3, 8, 9, 19, 14], tackling new and complex multi-agent problems that require decentralised solutions. One such problem is to combine planning and execution [13].

Multi-Agent Systems (MAS) development platforms also changed the focus from agent- to organisation-centred approaches. Recent research in MAS, as evidenced in [1, 16], shows that considering other programming dimensions such as environments and organisations as first-class abstractions (along with agents) allow developers to create more complex MAS.

Algorithm 1 DOMAP overview.
1: function DOMAP(Social_Goals)
2: for each agent ∈ Agents do
3: create own factored representation
4: end for
5: banned ← ∅
6: repeat
7: announce(Social_Goals, banned(List))
8: for each agent ∈ Agents do
9: for each sg ∈ Social_Goals do
10: if (agent, sg) /∈ banned then
11: agent calculates and places a bid for sg
12: end if
13: end for
14: end for
15: award(Social_Goals, bids)
16: // goalsagent are SGS allocated to agent
17: for each agent ∈ Agents do
18: individual_planning(goalsagent)
19: if planning_failed(goalsagent) then
20: for each sg ∈ goalsagent do
21: banned ← banned ∪ {(agent, sg)}
22: end for
23: end if
24: end for
25: until banned ≠ ∅
Figure 1: (a) Average plan cost across 10 runs; (b) Shortest parallel plan found across 10 runs.

3. EXPERIMENTS AND RESULTS

Figure 2: Average planning time across 10 runs.

We are grateful for the support given by CNPq and CAPES.

Acknowledgments

We are grateful for the support given by CNPq and CAPES.
REFERENCES


