Periodic Real-Time Resource Allocation for Teams of Progressive Processing Agents

Gilles S. Dibangoye, Abdel-Illah Mouaddib and Brahim Chaib-Draa

IFT-GLO - Laval University, Québec, Canada & GREYC - University of Caen, France

{gdibango, mouaddib}@info.unicaen.fr, brahim.chaib-draa@ift.ulaval.ca

Motivating Example

- **Nereus problem**: Naval anti-air warfare problem, depicted below.
- **Objectif**: Find a resource assignment strategy, that avoids both:
  1. The ship being hit by incoming enemy missiles;
  2. The bounded and shared resources being over-utilized.
- **Main difficulties posed by Nereus**:
  1. An arbitrary number of enemy missiles may occur over periods;
  2. The Lack of the joint-model of rewards and transitions of the team of enemy missiles;
  3. The large number of resources to allocate to the team of enemy missiles;
  4. The availability of resources positive and negative interactions;
  5. The resource constraints including tight response-time constraints.

Figure 1: Naval Environment for Resource Engagement in Unpredictable Situations.

Cyclic Progressive Reasoning Unit (C-PRU)

- **Model of Task Structure**:
  1. Each enemy missile is formalized as a structured task;
  2. Each task describes the ship engagement procedure as illustrated below.

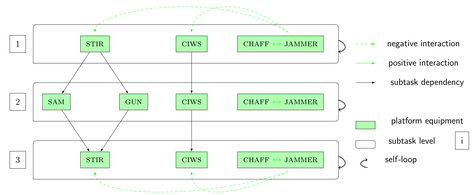


Figure 2: Cyclic Progressive Reasoning Unit of a missile engagement procedure.

- **More Formally**, a C-PRU C_i: $\langle S_i, A_i, T_i, R_i, K_i, U, \varphi, \lambda\rangle$ consists of:
  1. A single agent Markov decision problem: $\langle S_i, A_i, T_i, R_i, \lambda\rangle$;
  2. A resource management problem: $\langle K_i, U, \varphi\rangle$:
     - K_i is the set of resource types, such as CHAFF CLOUD or JAMMER as depicted above.
     - $\varphi(a_i, \Omega_i) \in [0, 1]$ is the discount factor of executing action a_i, when actions Ω_i are operating. Negative and positive interactions are illustrating in Figure above.
     - $U = [u_k]_k$ available amount of resources per type $k \in K_i$.
- **Remark**: C-PRU extends the progressive reasoning unit model first introduced by [Mouaddib & al.], for solving infinite-horizon MDPs handling structured tasks.

Two Phases Approximate Strategy

- **Inconvenient**: A periodic Multi-agent C-PRUs problem is PSPACE-HARD.
- **Solution**: Computing an approximate solution that sacrifices optimality for computational feasibility.
  1. **Offline phase**: Compute heuristic estimates of individual value function of each task:

$$\begin{aligned} q_i(s_i a_i|\Omega_i) &= R_i(s_i a_i|\Omega_i) + \lambda \textstyle\sum_{s_i'} T_i(s_i a_i s_i') V_i^\star(s_i'|\emptyset) \\ v_i(s_i|\Omega_i) &= \max_{a_i} q_i(s_i a_i|\Omega_i) \end{aligned} \tag{1}$$

  where $V^\star(s_i'|\emptyset)$ is an optimistic value when ignoring resource interactions, and

$$R_i(s_i a_i|\Omega_i) = \textstyle\sum_{s_i'} \varphi(a_i\Omega_i) T_i(s_i a_i s_i') R_i(s_i a_i s_i') \tag{2}$$

  2. **Online phase**: Recover an approximate estimate of the exact value function of the team of enemy missile tasks.

$$\bar{Q}(sa|\Omega) = \sum_{k=1}^{|M|} \zeta_k \prod_{i \le k} q_i(s_i a_i|\Omega_i) \tag{3}$$

  where ζ_k stands for the weight associated with task M_k, M is the set of all task M_k. The objective consists in finding a joint-action $\bar{a}^\star$ at each time period such that:

$$\bar{a}^\star = \arg\max_{a \in A} \bar{Q}(sa|\Omega) \tag{4}$$

- **Challenges**: To find a near-optimal solution through a search space bounded by $O(|\otimes_i A_i|^M)$ with respect to the response-time constraints: Each iteration improves the current complete solution.

Real-Time Dynamic A* (RTDA*)

- **Search Space**: Tree search structure $Tree = (N, E)$
  - $N = \{o_i^j\}_{i,j}$ denotes the set of nodes $o_i^j = \langle s_i, \Omega_i^j, Q_i^j\rangle$ describing agent M_i characteristics, where Q_i^j states the heuristic estimate of the current best joint action $\langle a_i \cdots a_{|M|}\rangle$:

$$Q_i^j = \textstyle\sum_{i'=i}^{|M|} \zeta_{i'} \prod_{i \le k \le i'} q_k(s_k a_k|\Omega_k^j) = q_i(s_i a_i|\Omega_i^j)(\zeta_i + Q_{i+1}^{j'}) \tag{5}$$

- **RTDA* Algorithm**:

Require: agent states $\{s^j\}_{j=1}^{|M|}$ ordered following EDF.
Ensure: *tree*.

```
tree ← EmptyTree
open ← EmptyStack
SearchSuccessors(o_1^0).
while open ≠ EmptyStack do
    ⟨o_i^j, a_i^{j'}, o_{i+1}^{j'}⟩ ← open.pop().
    if o_{i'}^{j+1} is not yet visited then
        add ⟨o_i^j, a_i^{j'}, o_{i+1}^{j'}⟩ to tree.
        SearchSuccessors(o_{i+1}^{j'}).
    else
        update Q_i^j as mentioned eq. (5).
    end if
end while
```

3 Agents Toy Example

- **Branch-And-Bound Algorithm**: RTDA* uses an upper U_b and lower L_b bounds to prune dominated nodes:

$$\begin{aligned} U_b(a_i^j) &= q_i[\zeta_i + v_{i+1}(\zeta_{i+1} + \zeta(|M| - i - 1))] \\ L_b(o_i^j) &= Q_i^j \end{aligned} \tag{6}$$

where $q_i = q_i(s_i a_i|\Omega_i^j)$, $v_i = v_i(s_i|\Omega_i^j)$ and $\zeta = \max_i \zeta_i$. As an example, action a_2^0 estimate value for $\zeta_i = \frac{1}{3}$ $(\forall i)$, is given by: $L_b = Q_2^0 = .34 \times (\frac{1}{3} + .44) = .2629$. The *upper bound* of selecting action a_2^1 (resp. a_2^2), is given by:

$$U_b(a_2^1) = .23 \times (\tfrac{1}{3} + .44 \times (\tfrac{1}{3} + \tfrac{1}{3}(3 - 2 - 1))) = .1104$$

Thus it is not necessary to expand either node o_3^1 or o_3^2 because of $U_b(a_2^1) < L_b$.

- **Anytime Algorithm**: RTDA* selects in a greedy fashion its joint-actions based on pre-compiled solutions.

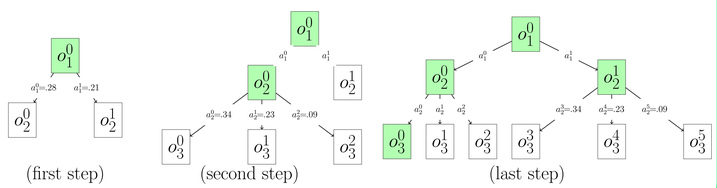
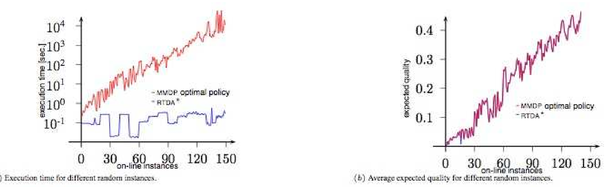


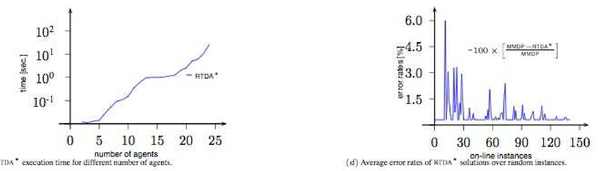
Figure 3: Each edge is labeled with an admissible action and its pre-compiled value. The light-green nodes mark the partially or completely explored node.

Experimental Results and Conclusion

- **Experiments**: Comparison of RTDA* and optimal MMDP solutions.



(a) Execution time for different random instances. (b) Average expected quality for different random instances.



(c) RTDA* execution time for different number of agents. (d) Average error rates of RTDA* solutions over random instances.

- **Results**: RTDA* is able to provide near-optimal solutions, with respect to bounded and shared resource, and under a large number of agents.
- **Future work directions**:
  1. Extend RTDA* in order to find optimal solutions;
  2. Adapt RTDA* for solving Dec-(PO)MDPs handling a large number of agents.