Theory of Potential Game Theoretic Control and Design of Movement Model

Yasuaki WASA

FL 11_05_01

20, May, 2011

Introduction (Cooperative Control)

Cooperative Control

Several autonomous agents connected with a network seek to collectively accomplish a global objective such as

Example: ・Sensor Coverage
・Consensus
・Resource Allocation

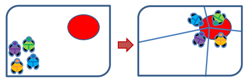


Fig.1 Coverage

Fig.2 Resource Allocation

Game Theoretic Approach

・Agents are "self-interested" ➡ Non-Cooperative Game

・The solution to the problem = the equilibrium of the game

Advantages

・Robustness to failures and environmental disturbances

・Reduction of communication requirements

・Scalability ・Adaptability in real time

● Use Potential Game

Introduction (Objective)

Previous Work

Propose new learning algorithm for Potential Game (PG), Payoff-based Inhomogeneous Partially Irrational Play (PIPIP) [1]

Motivation

・Not to make Experimental Studies for PIPIP
(Experimental Method of coverage or consensus problem)
・Not to consider the robustness to environmental disturbances

Objective of this work

・To verify the validity of the theoretic result [1] by way of experiment
・To consider the applicability of PIPIP to other scenarios
・To consider the applicability to an environmental change

[1] 後藤, "ポテンシャルゲーム理論的協調制御における学習アルゴリズムの提案," 東京工業大学修士論文, 2011.

Outline

- Introduction
- Essence of Potential Game
  - What's Potential Game?
  - Learning Algorithms (RSAP / PIPIP)
  - Problem of PIPIP
- Experimental Environment Settings
  - Movement Algorithm
  - Field Components
- Simulation / Experiment
- Summary and Future Work

Definition of Game Theory (Strategic game)

1. A set of players (agents) $\mathcal{V}=\{1,\cdots,N\}$ $(N\geq 2)$
   $(\mathcal{P}=\{\mathcal{P}_1,\cdots,\mathcal{P}_N\})$
2. The collection of action set $\mathcal{A}, a$
   $\mathcal{A}_i$: $\mathcal{P}_i$'s action set $\quad \mathcal{A}=\mathcal{A}_1\times\cdots\times\mathcal{A}_N$
   $a_i(\in\mathcal{A}_i)$: $\mathcal{P}_i$' s action $\quad a=(a_i,a_{-i})$
   $a_{-i}=(a_1,a_2,\cdots,a_{i-1},a_{i+1},\cdots,a_n)$
3. The collection of objective function $U=\{U_1,\cdots,U_N\}$
   U_i : $\mathcal{P}_i$' s objective function $\quad U_i:\mathcal{A}\rightarrow\mathbb{R}$
   Each agent chooses an action a_i to maximize U_i

Strategic Game is represented by $\Gamma=\langle\mathcal{V},\mathcal{A},U\rangle$

[Definition] (Pure Nash Equilibrium)

a pure Nash Equilibrium is an action profile $a^*\in\mathcal{A}$ s.t.

$$\forall i\in\mathcal{V}\quad U_i(a_i^*,a_{-i}^*)=\max_{a_i\in\mathcal{A}_i}U_i(a_i,a_{-i}^*)$$

Potential Game

Potential Function (global objective function) $\phi:\mathcal{A}\rightarrow\mathbb{R}$

ϕ : maximal ➡ objective as a group is achieved

Each agent chooses an action a_i to maximize U_i

[Definition] (Potential Game)

The strategic game $\Gamma=\langle\mathcal{V},\mathcal{A},U\rangle$ is a potential game if $\exists\phi$ s.t. $\forall i\in\mathcal{V}$ $\forall a_{-i}\in\mathcal{A}_{-i}$ $\forall a_i',a_i''\in\mathcal{A}_i$

$$U_i(a_i'',a_{-i})-U_i(a_i',a_{-i})=\phi(a_i'',a_{-i})-\phi(a_i',a_{-i})$$

Features of Potential Game

・Nash Equilibrium (NE) exists
・Design which leads to NE is easy
1. policies of objective function
2. Learning Algorithm

Ex.

Payoff

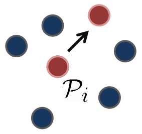
Player1 \ Player2	A	B
A	(2,2) NE	(1,0)
B	(0,1)	(4,4) NE

Potential

Player1 \ Player2	A	B
A	2	0
B	0	3

maximize ϕ = Optimal NE ⇄ (✕) NE

Restrictive Spatial Adaptive Play (RSAP)



Step1 Randomly choose one player $\mathcal{P}_i$

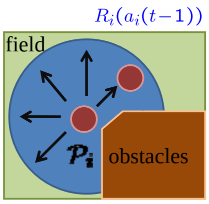
Step2 $\mathcal{P}_i$ selects one trial action $\hat{a}_i$

$\hat{a}_i \in R_i(a_i(t-1))$: restricted action set

$\Pr[\hat{a}_i = a_i] = 1/z_i$ $\quad z_i := \max_{a_i \in \mathcal{A}_i} |R_i(a_i)|$

$\forall a_i \in R_i(a_i(t-1)) \backslash a_i(t-1)$

$\Pr[\hat{a}_i = a_i(t-1)] = 1-((|R_i(a_i(t-1))|-1)/z_i)$



Step3 $\mathcal{P}_i$ choose its action $a_i(t)$: $\hat{a}_i$ or $a_i(t-1)$

Choose trial action

$$\Pr[a_i(t) = \hat{a}_i] = \frac{\exp\{\beta U_i(\hat{a}_i, a_{-i}(t-1))\}}{D}$$

Choose stay (previous) action

$$\Pr[a_i(t) = a_i(t-1)] = \frac{\exp\{\beta U_i(a(t-1))\}}{D}$$

$$D = \exp\{\beta U_i(\hat{a}_i, a_{-i}(t-1))\} + \exp\{\beta U_i(a(t-1))\}$$

Payoff-based Inhomogeneous Partially Irrational Play (PIPIP)[1]

I.C. $t = 0$: $a_i(0)$, $\forall i$ (randomly) ⇒ $U_i(a(0))$

$t = 1$: $a(1) = a(0)$, $U_i(a(1)) = U_i(a(0))$

Step1 $t \geq 2$: Update exploration rate : $\varepsilon(t) = t^{(-1/N(D+1))}$ Time-Inhomogeneous

Step2 **Action Selection** a_i^{tp}

case1 $U_i(a(t-1)) \geq U_i(a(t-2))$

probability	selection
$\varepsilon(t)$	$a_i^{tp} \in R_i(a_i(t)) \backslash \{a_i(t-1)\}$ randomly, uniformly
$1-\varepsilon(t)$	$a_i^{tp} = a_i(t-1)$

case2 otherwise $\Delta_i := U_i(a(t-2)) - U_i(a(t-1))$ $\quad k \in \left(\frac{1}{|C|-1}, \frac{1}{2}\right]$

probability	selection
$\varepsilon(t)$	$a_i^{tp} \in R_i(a_i(t)) \backslash \{a_i(t-1), a_i(t-2)\}$ randomly, uniformly
$(1-\varepsilon(t))(k\varepsilon(t)^{\Delta_i})$	$a_i^{tp} = a_i(t-1)$ ← an irrational decision
$(1-\varepsilon(t))(1-k\varepsilon(t)^{\Delta_i})$	$a_i^{tp} = a_i(t-2)$

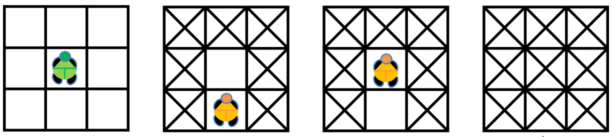
Step3 Executes the action a_i^{tp}

Step4 Compute $U_i(a_i^{tp}, a_{-i}^{tp}), R_i(a_i^{tp})$. Go to Step1

Problem of PIPIP

case2 otherwise

probability	selection
$\varepsilon(t)$	$a_i^{tp} \in R_i(a_i(t)) \backslash \{a_i(t-1), a_i(t-2)\}$ randomly, uniformly



$R_i(a_i)$ $\quad a_i(t-2)$ $\quad a_i(t-1)$ $\quad a_i^{tp}$

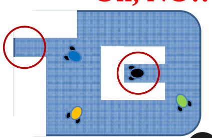
Oh, NO!!

In this case, this action does not exist.

(Drawback of PIPIP)

→ Unconsiderable case?

Or Not to satisfy Assumption ?



Reversibility and Feasibility are satisfied,

but for $a_i^{tp} \in R_i(a_i(t)) \backslash \{a_i(t-1), a_i(t-2)\}$?

Revised PIPIP

Step2 **Action Selection** a_i^{tp}

case1 $U_i(a(t-1)) \geq U_i(a(t-2))$

probability	selection
$\varepsilon(t)$	$a_i^{tp} \in R_i(a_i(t-1)) \backslash \{a_i(t-1)\}$ randomly, uniformly
$1-\varepsilon(t)$	$a_i^{tp} = a_i(t-1)$

case2 otherwise $\Delta_i := U_i(a(t-2)) - U_i(a(t-1))$ $\quad k \in \left(\frac{1}{|C|-1}, \frac{1}{2}\right]$

(i) $a_i^{tp} \in R_i(a_i(t-1)) \backslash \{a_i(t-1), a_i(t-2)\} \neq \emptyset$

probability	selection
$\varepsilon(t)$	$a_i^{tp} \in R_i(a_i(t-1)) \backslash \{a_i(t-1), a_i(t-2)\}$ randomly, uniformly
$(1-\varepsilon(t))(k\varepsilon(t)^{\Delta_i})$	$a_i^{tp} = a_i(t-1)$ ← an irrational decision
$(1-\varepsilon(t))(1-k\varepsilon(t)^{\Delta_i})$	$a_i^{tp} = a_i(t-2)$

(ii) $a_i^{tp} \in R_i(a_i(t-1)) \backslash \{a_i(t-1), a_i(t-2)\} = \emptyset$

probability	selection
$k\varepsilon(t)^{\Delta_i}$	$a_i^{tp} = a_i(t-1)$ ← an irrational decision
$1-k\varepsilon(t)^{\Delta_i}$	$a_i^{tp} = a_i(t-2)$

Outline

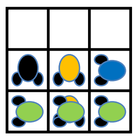
- Introduction
- Essence of Potential Game
  - What's Potential Game?
  - Learning Algorithms (RSAP / PIPIP)
  - Problem of PIPIP
- Experimental Environment Settings
  - Movement Algorithm
  - Field Components
- Simulation / Experiment
- Summary and Future Work

Concept of Movement Algorithm

Approach

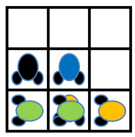
1. (agent on PG $\mathcal{P}_i$)= (machine i)
   - Intuitive understanding
   - Movement Algorithm is hard

   (especially, when multi-agent moves automatically)



2. (agent on PG $\mathcal{P}_i$) ≠ (machine i)
   - Intuitive understanding is hard?

   (to consider machine ability, sensor ability ...)
   - Movement Algorithm is easy
   - "Nearest Agent" moves



Approach 2 is better in the Coverage Problem

In the Consensus Problem … ?

Movement Algorithm of Approach 2

I.C. $t = 0$: get $a(0), \hat{a}(0)$ and set machine numbers $p_i(0) = i, \forall i \in \mathcal{V}$

Step1 $t \geq 1$: get next state $\hat{a}(t)$ (*)

Step2 For all combination : ($N!$ patterns)

(i) Correspond Machine i to agent on PG j :

trial number : $\hat{p}_i(t) = j,\ i, j \in \mathcal{V}$ $(\hat{p}_i(t) \neq \hat{p}_j(t),\ i, j \in \mathcal{V})$

(ii) Compute the Estimate Function J :

$$J = \sum_{i \in \mathcal{V}} \|\hat{a}_{\hat{p}_i(t)}(t) - \hat{a}_{p_i(t-1)}(t-1)\|_1$$

Step3 $p(t) = \hat{p}(t)$ s.t. $\min J$

Step4 Each agent computes "route step" (Dijkstra's algorithm)

Step5 **Collision Avoidance**

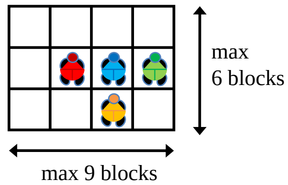
(A) A head-on collision : change $p(t)$. Back to Step2.

(B) other collision : one machine waits several steps

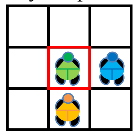
Step6 Each agent execute the actions

Step7 Go to Step1

Field Components (Coverage)



If several agents stay at the same position, they are put at neighbors



(*) To decide $\hat{a}(t)$

I.C. $t = 0$: All agent are set not to be repeated $\hat{a}(0)(= a(0))$

Step1 Set the position where an agent exists.

Not to be repeated : Go to Step3. Otherwise : Go to Step2.

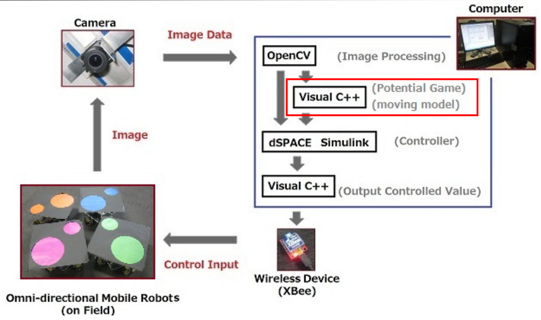
Step2 Search the neighbor place where an agent is not set

[Condition] ・Reachable from the place where the agent exists

・movable in the shortest time from the current position

Step3 Give the next state $\hat{a}(t)$

Experimental Environment



Pose Synchronization Law to the Imaginary Pose[4]

$$\begin{bmatrix} v^b_{xi} \\ v^b_{yi} \\ \omega^b_i \end{bmatrix} = \begin{bmatrix} k_{pi} & 0 & 0 \\ 0 & k_{pi} & 0 \\ 0 & 0 & k_{ai} \end{bmatrix} \begin{bmatrix} \cos\theta_{wi} & \sin\theta_{wi} & 0 \\ -\sin\theta_{wi} & \cos\theta_{wi} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{wri} - x_{wi} \\ y_{wri} - y_{wi} \\ \sin(\theta_{wri} - \theta_{wi}) \end{bmatrix}$$

Outline

- Introduction
- Essence of Potential Game
  - What's Potential Game?
  - Learning Algorithms (RSAP / PIPIP)
  - Problem of PIPIP
- Experimental Environment Settings
  - Movement Algorithm
  - Field Components
- Simulation / Experiment
- Summary and Future Work

Coverage : Settings

Area $Q = \{1, \cdots, 9\} \times \{1, \cdots, 6\}$

Sensing Area $D(a_i)$ Radius to 1

Reward (Gaussian Distribution)

$$W_q(q) = \exp\left\{-\frac{(q_x - 7)^2 + (q_y - 5)^2}{8}\right\}$$

Obstacle

$$\mathcal{O} = \{(x, y) | x + y = 8,\ 3 \leq x \leq 6\}$$

Agents 4

Initial Position

Simulation All agents : (2,2)

Experiment (1,1), (1,2), (2,1), (2,2)

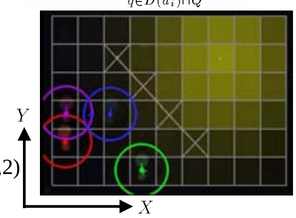
Skips (Experiment) 10 steps

Potential Function

$$\phi(a) = \sum_{q \in Q} \sum_{l=1}^{n_q(a)} \frac{W_q}{l}$$

Utility Function

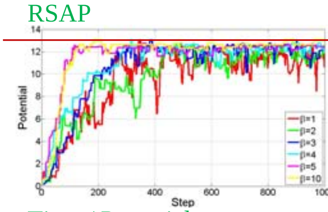
$$U_i(a_i, a_{-i}) = \sum_{q \in D(a_i) \cap Q} \frac{W_q}{n_q(a)}$$

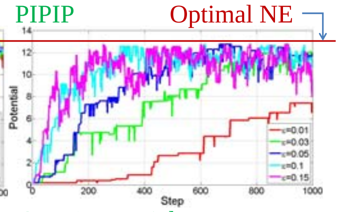


(to take a lot of time) 100steps⇒12minutes, at least 1000steps …

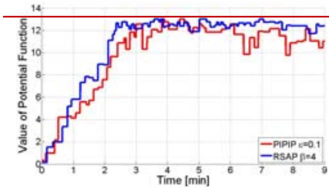
Plain Field

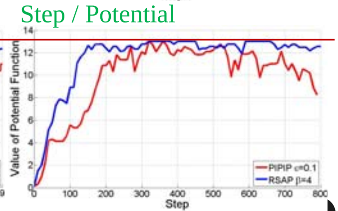
RSAP PIPIP Optimal NE



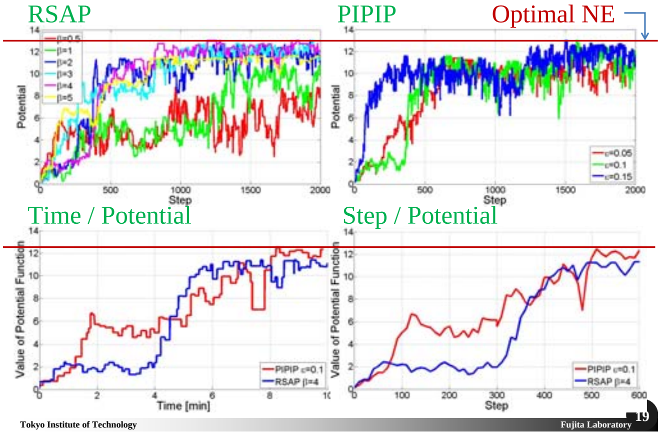


Time / Potential Step / Potential





Obstacle



Outline

- Introduction
- Essence of Potential Game
  - What's Potential Game?
  - Learning Algorithms (RSAP / PIPIP)
  - Problem of PIPIP
- Experimental Environment Settings
  - Movement Algorithm
  - Field Components
- Simulation / Experiment
- Summary and Future Work

Summary and Future Work

Summary

- Compose Experimental System for PG
- Verify the theoretic result [1] by way of experiment
- Compare RSAP and PIPIP in the Consensus problem

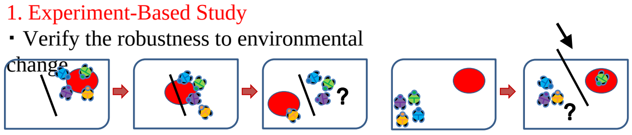
Next Seminar

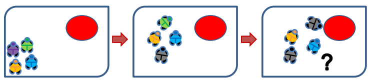
Future Works

1. Experiment-Based Study
   - Verify the robustness to environmental change
   - Tackle Coverage Problem with camera
   - Compare RSAP and PIPIP in the Attitude Coordination
2. Simulation-Based Study
   - Compose Simulator (Simulation→Movie)
   - Apply to a large system
     (variables : field, agents, sensing area, reward etc.)

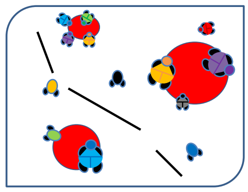
Details of Future Work

1. Experiment-Based Study
- Verify the robustness to environmental change

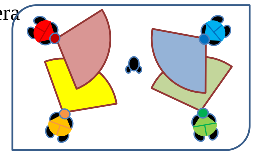




2. Simulation-Based Study
- Apply to a large system



- Tackle Coverage Problem with camera



Reference

[1] 後藤, "ポテンシャルゲーム理論的協調制御における学習アルゴリズムの提案," 東京工業大学修士論文, 2011.

[2] J. R. Marden, G. Arslan and J. S. Shamma, "Cooperative Control and Potential Games," *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 39, No. 6, pp. 1393-1407, 2009.

[3] M. Zhu and S. Martinez, "Distributed Coverage Games for Mobile Visual Sensors (i): Reaching the Set of Nash Equilibria," *Proc. of the 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pp. 169-174, 2009.

[4] 伊吹, "車輪型移動ロボットを用いた位置姿勢同期制御に関する研究," 東京工業大学学士論文, 2008.

APPENDIX

- The Law of Local Controller
- Global Function
- Distributed Inhomogeneous Synchronous Learning (DISL)
- RSAP v.s. PIPIP
- Field Components
- Dijkstra's Algorithm

The Law of Local Controller

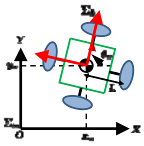
Pose Synchronization Law[4]

$$\begin{bmatrix} v_{xi}^b \\ v_{yi}^b \\ \omega_i^b \end{bmatrix} = \begin{bmatrix} k_{pi} & 0 & 0 \\ 0 & k_{pi} & 0 \\ 0 & 0 & k_{ai} \end{bmatrix} \sum_{j \in \mathcal{N}_i} \begin{bmatrix} \cos\theta_{wi} & \sin\theta_{wi} & 0 \\ -\sin\theta_{wi} & \cos\theta_{wi} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{wj} - x_{wi} \\ y_{wj} - y_{wi} \\ \sin(\theta_{wj} - \theta_{wi}) \end{bmatrix}$$

Pose Synchronization Law to the Imaginary Pose

$$\begin{bmatrix} v_{xi}^b \\ v_{yi}^b \\ \omega_i^b \end{bmatrix} = \begin{bmatrix} k_{pi} & 0 & 0 \\ 0 & k_{pi} & 0 \\ 0 & 0 & k_{ai} \end{bmatrix} \begin{bmatrix} \cos\theta_{wi} & \sin\theta_{wi} & 0 \\ -\sin\theta_{wi} & \cos\theta_{wi} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{wri} - x_{wi} \\ y_{wri} - y_{wi} \\ \sin(\theta_{wri} - \theta_{wi}) \end{bmatrix}$$

$[\, x_{wri} \;\; y_{wri} \;\; \theta_{wri} \,]^T$: reference pose to lead PG movement algorithm



[4] 伊吹, "車輪型移動ロボットを用いた位置姿勢同期制御に関する研究," 東京工業大学学士論文, 2008.

Global Function

Coverage

Potential Function

$$\phi(a) = \sum_{q \in Q} \sum_{l=1}^{n_q(a)} \frac{W_q}{l}$$

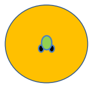
Utility Function

$$U_i(a_i, a_{-i}) = \sum_{q \in D(a_i) \cap Q} \frac{W_q}{n_q(a)}$$

Settings

Reward $W_q(q)$

Sensing Area $D(a_i)$



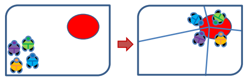


Fig.1 Coverage

Consensus

Potential Function

$$\phi(a) = -\sum_{\mathcal{P}_i \in \mathcal{P}} \sum_{\mathcal{P}_j \in N_i} \frac{\|a_i - a_j\|}{2}$$

Utility Function

$$U_i(a_i, a_{-i}) = -\sum_{\mathcal{P}_j \in N_i} \|a_i - a_j\|$$

Settings

Communication N_i

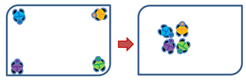


Fig.2 Consensus

Distributed Inhomogeneous Synchronous Learning (DISL)

I.C. $t = 0$: $a_i(0)$, $\forall i$ randomly select and compute $U_i(a(0))$

$t = 1$: $a(1) = a(0)$, $U_i(a(1)) = U_i(a(0))$

Step1 $t \geq 2$: Each agent updates exploration rate as $\varepsilon(t) = t^{(-1/N(D+1))}$

Compute $a_i(\tau_i(t))$ Time-Inhomogeneous

N : number of agents

D_i : minimum step of any two different action $D := \max_{i \in \mathcal{V}} D_i$

$$\tau_i(t) = \begin{cases} t-1 & \text{if } U_i(a(t-1)) \geq U_i(a(t-2)) \\ t-2 & \text{otherwise} \end{cases}$$

Convergence Nash Equilibrium

Step2 **Action Selection** a_i^{tp}

probability	selection	
$\varepsilon(t)$	$a_i^{tp} \in R_i(a_i(t-1)) \backslash \{a_i(\tau_i(t))\}$	randomly, uniformly
$1 - \varepsilon(t)$	$a_i^{tp} = a_i(\tau(t))$	

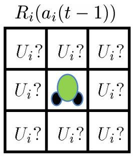
Step3 Each agent executes the action a_i^{tp}

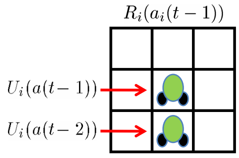
Step4 Compute $U_i(a_i^{tp}, a_{-i}^{tp}), R_i(a_i^{tp})$. Go to Step1

Problem This does not imply that the group achieves optimal Nash equilibria ➡ PIPIP

RSAP v.s. PIPIP

	RSAP	PIPIP
Memory	Finite	Finite
Information	Virtual Payoff-based	Payoff-based
	$R_i(a_i(t-1))$	$R_i(a_i(t-1))$

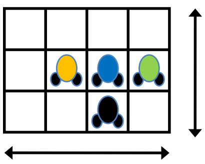




	RSAP	PIPIP
Exploration Parameter	$\beta \geq 0$	$\varepsilon \in (0, 1/2]$
Convergence	faster $(\beta \to \infty)$	faster $(\varepsilon \to 1/2)$
Exploration rate	lower $(\beta \to \infty)$	higher $(\varepsilon \to 1/2)$
Movable agent	One agent	All agents
Equilibrium	Optimal NE (high probability)	Optimal NE (probability 1)

Field Components

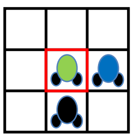
Coverage



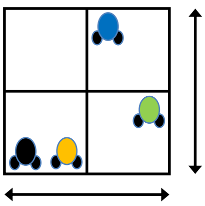
max 6 blocks

max 9 blocks

If several agents stay at the same position, they are put at neighbors



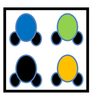
Consensus



max 3 blocks

max 4 blocks

Stay Position



Dijkstra's Algorithm

Settings Network $N = [\mathcal{G} = (\mathcal{V}, \mathcal{E}); s, d]$

$\mathcal{G}$: Connected, No-direction graph

$s \in \mathcal{V}$: Start position

$d : \mathcal{E} \to \mathbb{R}_+$: length

$$U := \{s\},\; T := \emptyset,\; u := s,\; d^*(s) := 0,\; d^*(v) := \infty \; (v \in \mathcal{V} - \{s\})$$

Step While $\mathcal{V} - U \neq \emptyset$

(i) For all sides $\{u, v\} \in \mathcal{E}$ to connect u and $v \in \mathcal{V} - U$

$$d^*(v) := \min\{d^*(v), d^*(u) + d(u, v)\}$$

If $d^*(v)$ is updated, set $e^*(v) := (u, v)$

(ii) For $(u, v_{\min})$ s.t. $\min_{v \in \mathcal{V} - U} d^*(v)$

$$U := U \cup \{v_{\min}\},\quad T := T \cup \{e^*(v_{\min})\}$$

If $\mathcal{V} - U \neq \emptyset$, set $u := v_{\min}$

Output $T = \{e^*(v) | v \in \mathcal{V} - \{s\}\}$

Dijkstra's Algorithm can lead the minimum root