Coordination dans les SMA

Problématique et Approches

AMAL EL FALLAH SEGHROUCHNI

Amal.Elfallah@lip6.fr

Rappel

RÉSUMÉ DU COURS 1

Pourquoi les SMA ?

- Efficacité
  - Décomposition et répartition des connaissances et des traitements (Unité de base : agent)
  - Aptitude à traiter des problèmes simultanés et potentiellement corrélés avec des optimisations éventuelles
- Souplesse
  - Adaptabilité et possibilité d'apprentissage des agents
  - Résistance à des environnements évolutifs ou instables
- Paradigme de conception
  - Adéquation à des applications distribuées et coopératives
  - Systèmes ouverts et intégration d'opérateurs humains
  - Modularité, réutilisabilité, interopérabilité, etc.

Apports des SMA

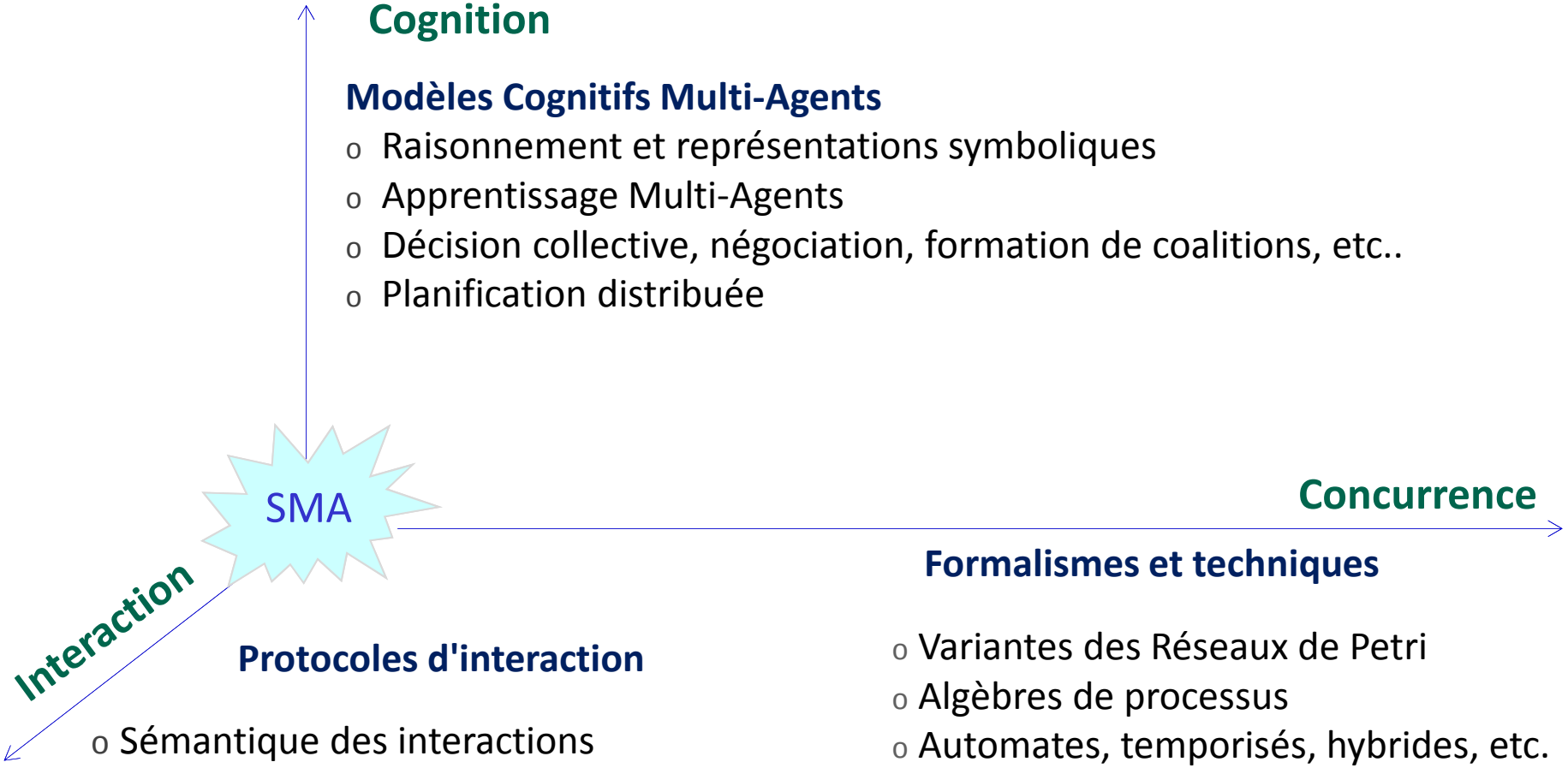
- Conception de systèmes complexes
  - Hétérogénéité des agents
  - Distribution géographique ou des traitements
  - Grande quantité d'information
  - Pas de solution algorithmique simple
  - Problème de nature heuristique
  - Approche basée sur l'interaction (faire communiquer plusieurs systèmes intelligents)

- Modélisation de systèmes hybrides, ouverts ...
  - Agents coopératifs et/ou compétitifs
  - Agents cognitifs et/ou réactifs

Positionnement des SMA

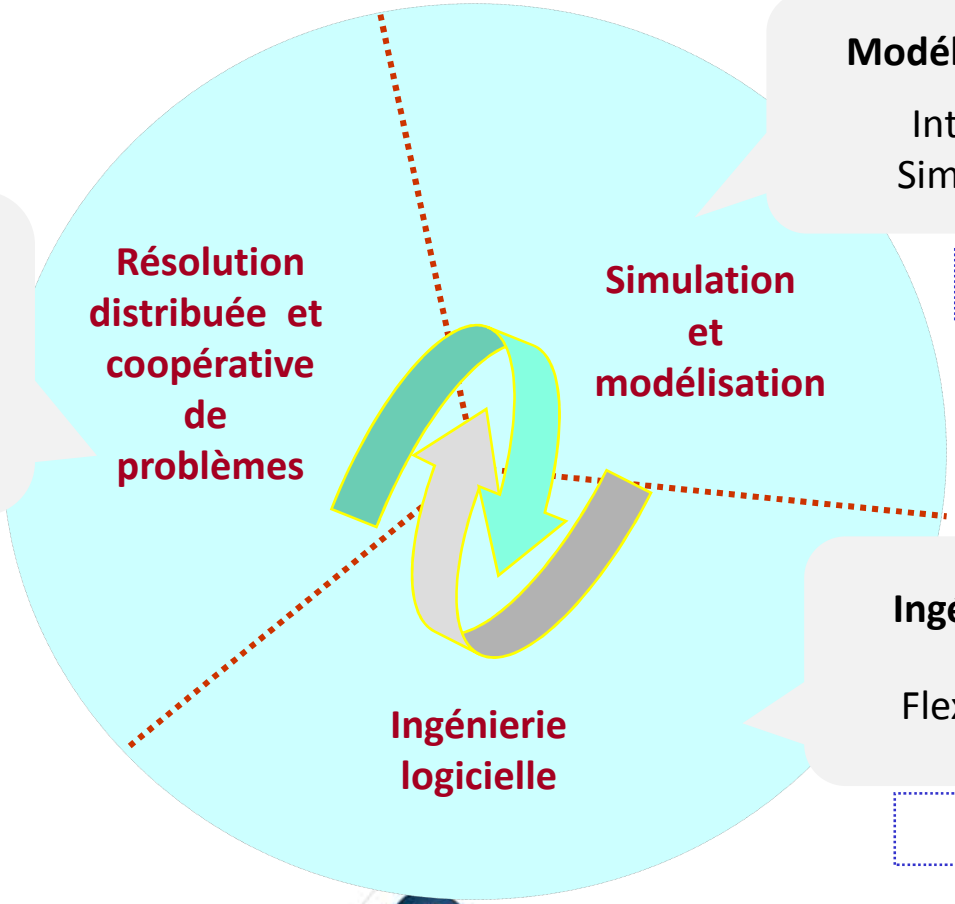
- Le domaine SMA vient de l'IA
  - → Domaine relativement général
  - → Fédère plusieurs domaines techniques de l'IA (planification, coordination, négociation, décision, apprentissage)
- Recoupe l'informatique distribuée
- Rejoint le génie logiciel
- Le domaine SMA porte son attention sur les systèmes :
  - Hétérogènes
  - Ouverts
  - Mixtes (machines et humains)

Espace SMA cognitifs



SMA : 3 classes de problèmes scientifiques

[Action Spécifique du CNRS : SMA, 2004-2005]



Approche cognitive
(ex. multi-experts)

Approche réactive
(ex : eco-systèmes)

Intelligence Artificielle distribuée

Modélisation et simulation

Interactions, environment
Simulation individu-centrée

Socio-mimétique

Ingénierie orientée Agent

Flexibilité + Coopération

Ingénierie logicielle

Plan

1. Problématique de la coordination
2. Approches de coordination
3. Coordination par planification distribuée
4. Coordination par formation de coalitions
5. Conclusion

Pourquoi la coordination ?

- Hypothèses
  - Agents logiciels, autonomes, hétérogènes, coopératifs ou concurrents, distribués et communicants
  - Environnement partagé et dynamique
  - Ressources limitées

- Problèmes
  - Génériques
    - Exécutions concurrentes => Blocage, famine, etc.
    - Communications asynchrones => Absence d'état global
  - Spécifiques
    - Définition des communications dans un système ouvert
    - Gestion des informations incomplètes, vision partielle des agents
    - Conciliation entre autonomie et exécution globale

Problématique de la coordination

- Différentes perspectives
  - Problème de prise de décision distribuée sous incertitude
  - Problème relatif à la connaissance et l'action
  - Problème de conception tel que la coordination de "frameworks" pour des organisations Homme-Machine
  - Etc.
- Processus qui contrôle et/ou guide le comportement des agents
  - Atteindre ou maintenir un état global où les états locaux des agents sont compatibles (résolution d'interactions négatives)
  - Favoriser la synergie des agents (résolution d'interactions positives)

Exemple

- Rationalité économique
  - Objectifs propres
  - Égoïsme
  - Pas de tendance à la coopération (sauf si bénéfique)
- Forte autonomie
  - Interventions humaines potentielles
  - Possibilité de triche (anonymat)
- Hétérogénéité
  - Conception décentralisée
  - Conception diffuse dans le temps
- Exemple de compagnies aériennes

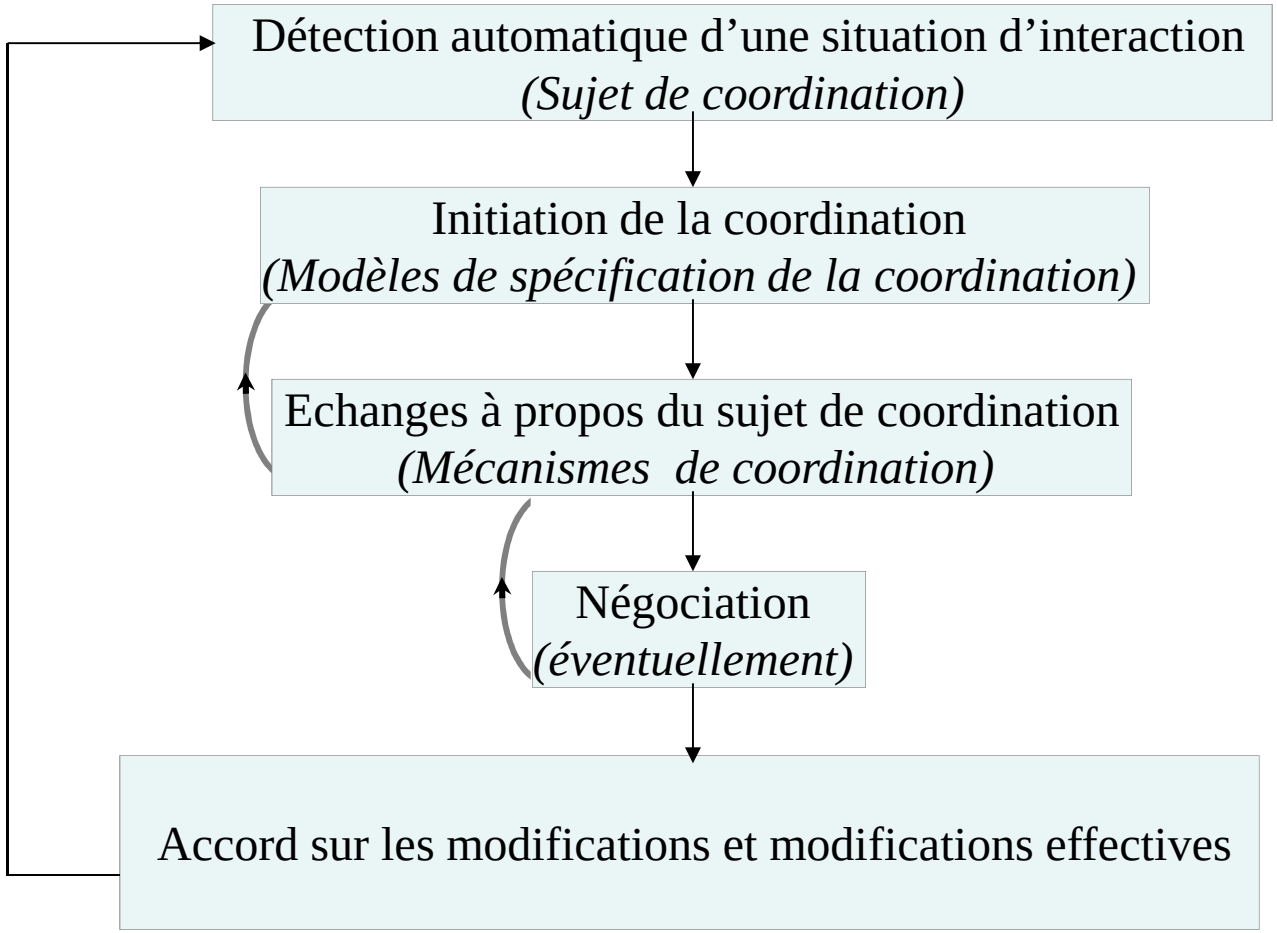
Modèles de coordination (1/2)

- Modèles orientés tâches
  - Issus de la résolution distribuée de problèmes
    - Existence d'un agent « central »
    - Affectation des tâches et coordination souvent centralisées
    - Existence d'un but global
    - Agents généralement coopératifs
    - Optimisation de l'efficacité globale
- Exemple
  - Scénario « Transport de marchandises »
    - Agents Convoyeurs/Marchants coopératifs
    - Rôles répartis et tâches pré-spécifiées

Modèles de coordination (2/2)

- Modèles orientés agents
  - Absence de but global
  - Absence d'agent « central »
  - Coordination distribuée (ex. négociation, consensus, etc.)
  - Agents généralement compétitifs
    - Exemple : optimisation d'une fonction d'utilité individuelle
- Exemple
  - Scénario du «Commerce électronique»
    - Agents égoïstes : maximisation du profit individuel
    - Coordination basée sur l'atteinte de consensus
    - Stratégies individuelles inconnues *a priori*

Principales phases d'un mécanisme de coordination



Principales approches

- Coordination orientée résolution distribuée de problèmes
  - Distributions spatiales, fonctionnelles, temporelles
  - Ex. DVMT (PGP et GPGP)
- Coordination basée sur les structures organisationnelles
  - Organisations statiques versus dynamiques
  - Exemple : systèmes normatifs, systèmes de règles, etc.
- Coordination basée sur les protocoles de coopération
  - Coopération par interaction
  - Exemple : enchères, CNP, etc.
- Négociation et prise de décision distribuée
  - Théorie des jeux, théorie d'aide à la décision, etc.
- Coordination par planification multi-agent ou distribuée
- Coordination fondée sur la formation de coalitions

Coordination par planification distribuée

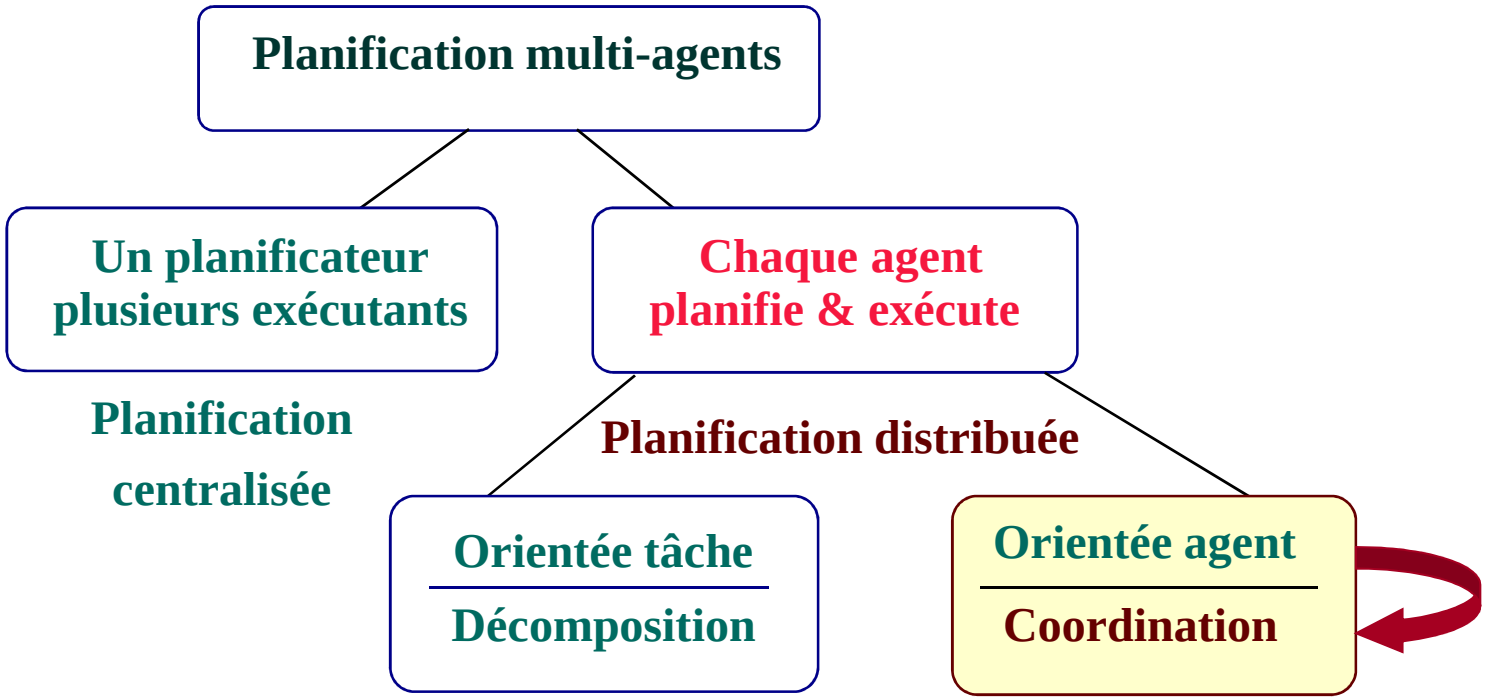
Plan : Abstraction du comportement d'un agent

Coordination des plans

Un plan ?

- Un plan est un ensemble d'actions ordonné totalement ou partiellement ; il permet à partir d'un état initial, d'atteindre un état satisfaisant le but du plan.

Planification multi-agents



Scénario : réseau de trafic maritime

- 2 classes d'agents : convoyeurs et marchands
  - Convoyeur transporte la marchandise
    - In_Hang : Dépot de marchandise
    - Out_Hang : Retrait de marchandise
    - Boat de volume limité
  - Marchand distribue la marchandise
    - Ne peut déposer que dans In_Hang d'un convoyeur
    - Ne peut retirer que d'un Out_Hang d'un convoyeur

- Agents situés sur des nœuds distincts du réseau
- Déplacement entre les nœuds via des liens dynamiques
  - Nœuds reliés par des canaux
  - Canaux peuvent être encombrés, momentanément fermés, etc.

Coordination par planification distribuée

- Principes du modèle proposé
  - Plan, abstraction du comportement de l'agent
  - Rôles des agents connus a priori (classes d'agents)
  - Coordination par échange et fusion de plans
  - Réaction à l'environnement par dynamicité d'exécution de plans

- Mécanismes d'exécution
  - Allocation dynamique des tâches
  - Actions abstraites et raffinement dynamique
  - Évaluation multi-critères des plans
  - Résolution des interactions par coordination de plans

Modèle récursif de planification

- Modèle formel de planification
  - Relations hiérarchiques entre plans
  - Relations entre actions (choix, concurrence, ordre, ..)
  - Raffinement dynamique (hiérarchie, récursivité)
  - Contraintes d'exécution d'action (pré- et post-conditions)
  - Autonomie, généricité mais, contrôle à l'exécution

- Théorie de coordination de plans
  - Ordonnancement consistant et allocation de ressources
  - Caractérisation, reconnaissance automatique et gestion des situations d'interactions : négatives et positives
  - Définition des méthodes de transformation des plans

Quel modèle de planification distribuée ?

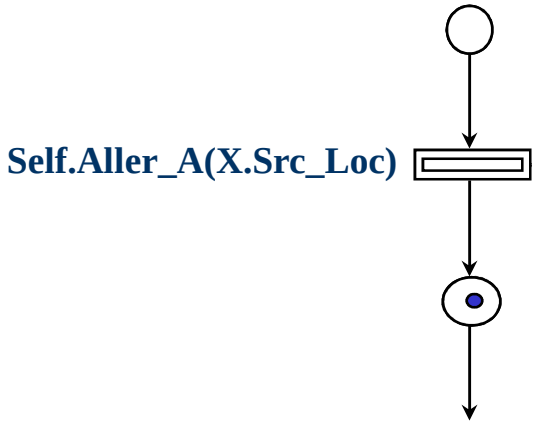
- Modèle basé sur les Réseaux de Petri récursifs (RdPR)
  - Orienté buts (déclaratif)
  - Unifie la résolution des interactions positives (synergie) et négatives (conflits)
  - Structure hiérarchique de plans (actions abstraites)
  - Dynamicité (raffinement)
  - Modèle théorique et interprété
- Algorithmes de coordination de plans
  - Opérations sur les plans (fusion, séquentialisation, parallélisation)
  - Vérification de la consistance de plans (graphe d'accessibilité)

Qu'est-ce qu'un RdPR ?

- Extension stricte des RdP introduisant la dynamicité dans la structure par le typage des transitions

  - Le franchissement d'une transition abstraite la déplie en un sous-réseau
  - Le franchissement d'une transition de fin replie le réseau déplié

  → L'état courant d'un RdPR est un arbre de réseaux marqués

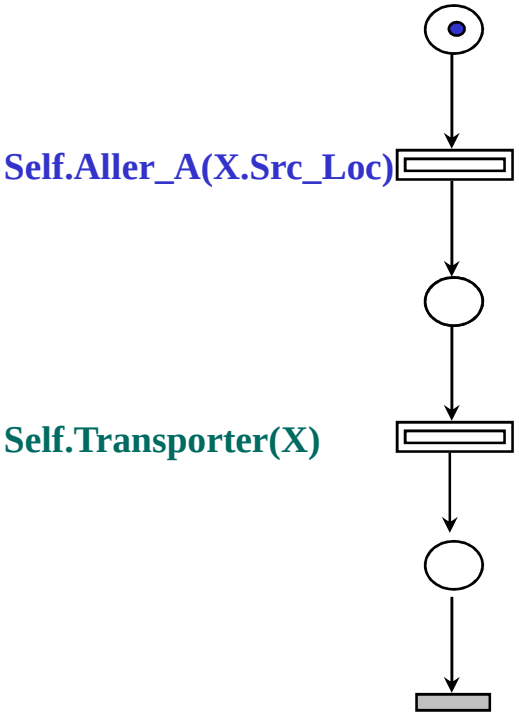


Modèle interprété

- Modèle théorique enrichi
  - Objets : modèle du monde
  - Méthodes : actions sur le monde
  - Variables du plan
  - Conditions sémantiques portant sur les objets du monde pour contrôler l'exécution (pré- et post-conditions d'actions)
- Un plan P est un RdPR interprété
  - P = <R, Met, Var>
    - R : squelette du plan
    - Met : ensemble de méthodes associées aux transitions abstraites ou élémentaires
    - Var : ensemble de variables.

→ Raffinement dynamique : planification conditionnelle

Exemple de Méthode



Plan du Convoyeur

Method		Ag.Aller_A(Dest)	
Type		Abstract	
Variables		Conditions	
Name	Class	Pre-	Post-
Ag	Convoyeur	none	Ag.Cur_Loc =Dest
Dest	Location	none	none

Une Méthode :

un label,

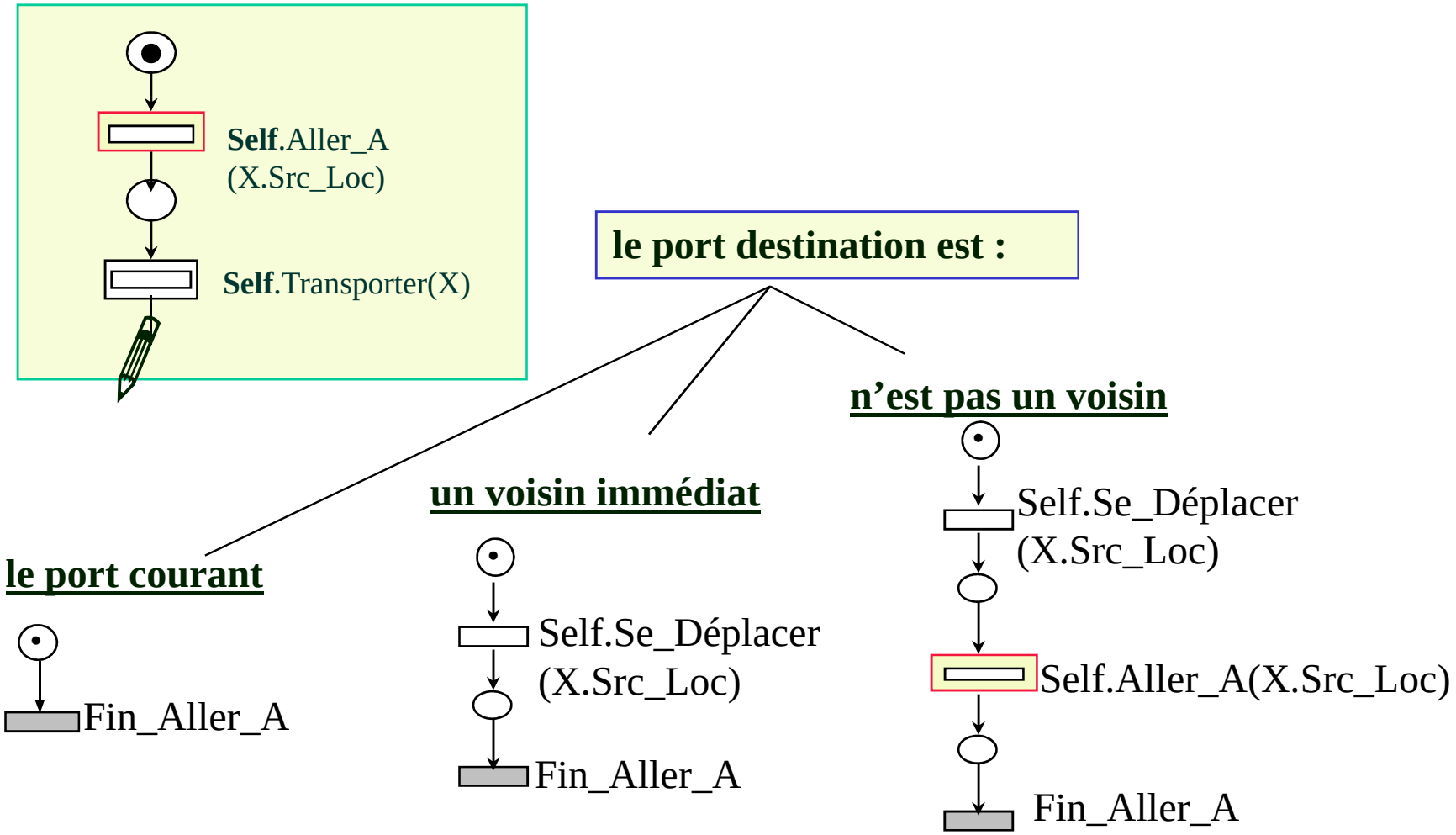
le type de la méthode

un ensemble de RdPR initialisés dans le cas d'une transition abstraite.

Dynamicité et Raffinements

- Plusieurs raffinements associés à une même transition abstraite
  - un raffinement est une méthode d'exécution possible
  - une méthode possible est un RdPR initialisé
  - Méthode = < R, M_0, B_0> où :
    - R : < P, T, W-, W+, Var, Call>
    - M_0 : marquage initial de R
    - B_0: fonction qui lie totalement ou partiellement les variables (Var) aux objets du domaine.

Raffinement dynamique : planification conditionnelle



Etat d'un RdPR : arbre de réseaux marqués

- Tr = < S, A>
  - $Tr_0 = (R_0 , M_0 , B_0)$
  - pour tout arc a = (s, s') / trans(a) = t
    - si t est une transition élémentaire :
      - mise à jour du marquage
    - si t est une transition abstraite :
      - on crée un nouveau sommet s dans Tr labellé par t
    - si t est une transition de fin :
      - Tr est élagué à partir du nœud courant et
      - le marquage est mis à jour dans l 'arbre d 'exécution

Sémantique du RdPR

- Condition de franchissabilité
  - Une transition *t* est franchissable à partir du nœud si et seulement si : $M(s) \geq W^-(., t)$

- Transition élémentaire
  - Soit t ∈ Telem d'un RdPR associé à un nœud s de Tr, le franchissement de t à partir de s produit l'arbre Tr' tel que :

  - Tr' = (STr' , ATr') avec :
    - STr = STr' tel que ∀ s' ≠ s, MTr'(s') = MTr(s') **(inchangé pour tout s' ≠ s),**
    - ATr = ATr'
  - $MTr'(s) = MTr(s) + W^+(., t) - W^-(., t)$

Sémantique des RdPR

- Transition abstraite
  - Soit t ∈ Tabs d'un RdPR associé à un nœud s de Tr, le franchissement de t à partir de s produit l'arbre Tr' tel que :
  - Tr' = (STr' , ATr') avec :
    - STr' = STr ∪ {s_n} où le RdPR associé au nouveau sommet s_n est l'un des réseaux implantant meth(t) et MTr'(s_n) = M_0 (marquage initial d'appel associé à ce RdPR) ,
    - ATr'= ATr ∪ {a} où a = <s, s_n> et transTr (a) = t ,
  - MTr'(s) = MTr(s) - W^-(., t).

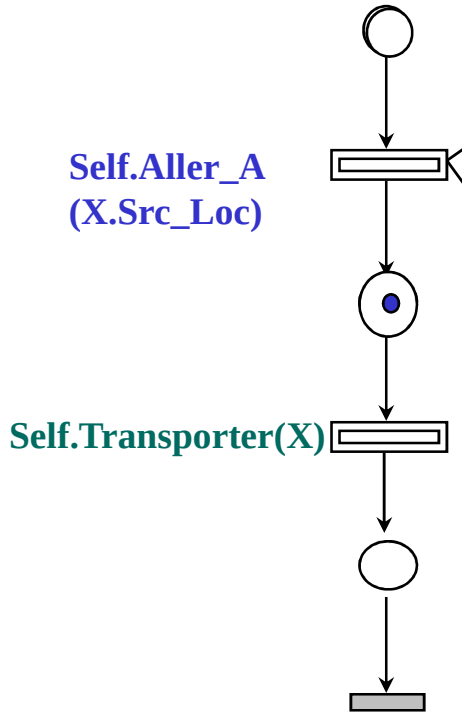
Sémantique des RdPR

- Transition de fin
  - Soit t ∈ Tfin d'un RdPR associé à un nœud s de Tr, le franchissement de t à partir de s produit l'arbre Tr' tel que :
  - Tr' = PRUNE(Tr, s) Soit s' le père de s dans l'arbre Tr et a = <s', s> l'arc étiqueté par t, alors :

    $\forall a \in ATr', trans\ Tr'(a) = transTr(a)$,

    $\forall s'' \in (STr' \setminus \{s'\}), MTr'(s'') = MTr(s'')$,
  - $MTr'(s') = MTr(s') + W^{+}(., transTr(a))$.

Exemple



Plan Convoyeur

$$S_0 = (R_0 , M_0 , B_0) , M_0 = (0,1,0)$$

Mécanismes de coordination

Situations de coordination

- Interactions positives entre plans
  - actions redondantes
    - détection statique : séquentialisation
  - actions favorables
    - détection dynamique : incorporation
- Interactions négatives entre plans
  - actions défavorables
  - actions exclusives
  - actions incompatibles

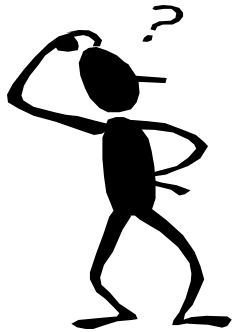
Algorithmes de coordination

- Fusion pour traitement d'interactions positives
  - Synchronisation par unification des transitions
- Fusion pour traitement d'interactions négatives
  - Ajout des places de synchronisation (séquentialisation)
  - Algorithme de parallélisation
- Vérification de la consistance du plan fusionné

Traitement d'une interaction positive

Contexte :

Agent incapable d'exécuter une action de son plan (initial ou raffiné)



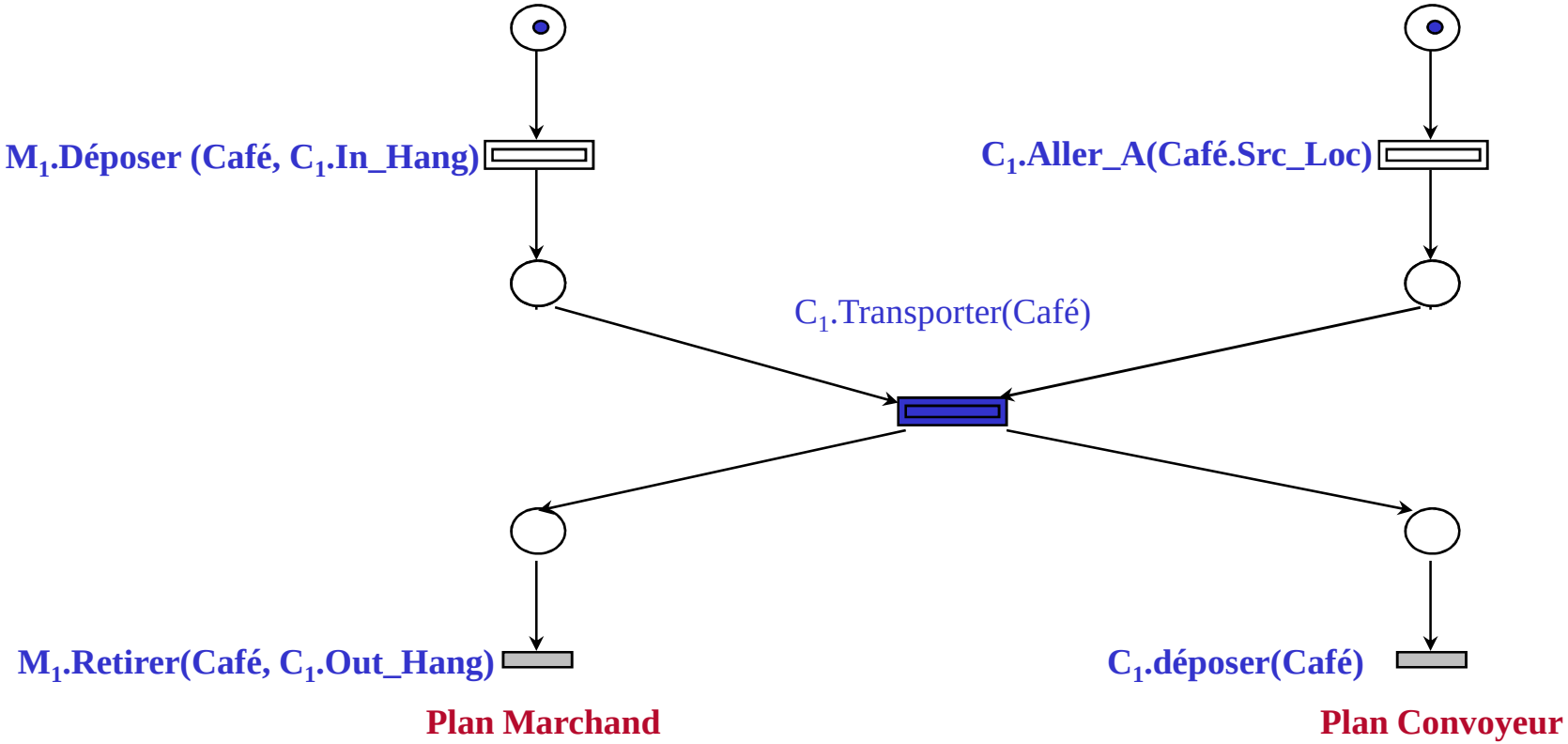
- **Initiation de la coordination**
  - **Recherche d'un agent à partir de**
    - ✓ liste d'accointances
    - ✓ les attributs de la variable non instanciée (Source, Type Agent, etc.)
  - **émission du plan pour coordination**

Traitement d'une interaction positive

- L'agent récepteur
  - Reconnaissance et unification
    - Unification et instanciation des variables
    - Fusion structurelle par transition commune
  - Développement de l'arbre d'accessibilité
    - Calcul des Pré- et Post-Conditions
    - Algorithme PPCC [ICMAS'96]

- Résultats
  - Le récepteur contrôle l'exécution
  - Exécutions parallèles Récepteur/Emetteur
  - Synchronisation via la transition commune

Synchronisation par unification des transitions communes



Validation de plan

- Validation du plan : *Adéquation* du modèle interprété au modèle non interprété, i.e.

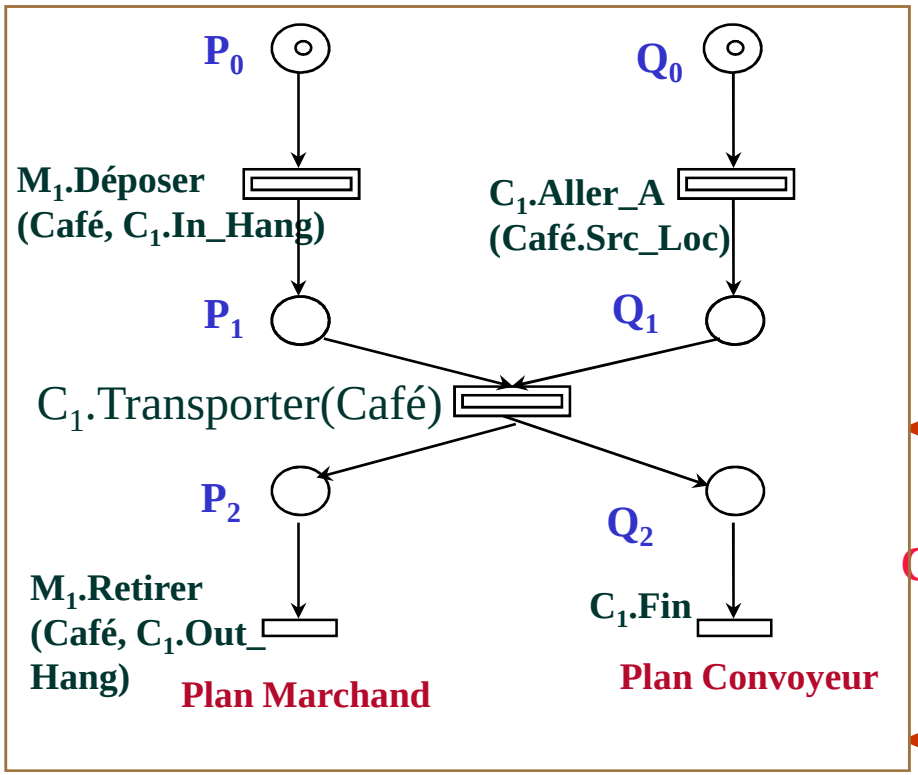
  toute séquence de franchissement valide dans le modèle non interprété est une séquence d'actions exécutable dans le modèle interprété

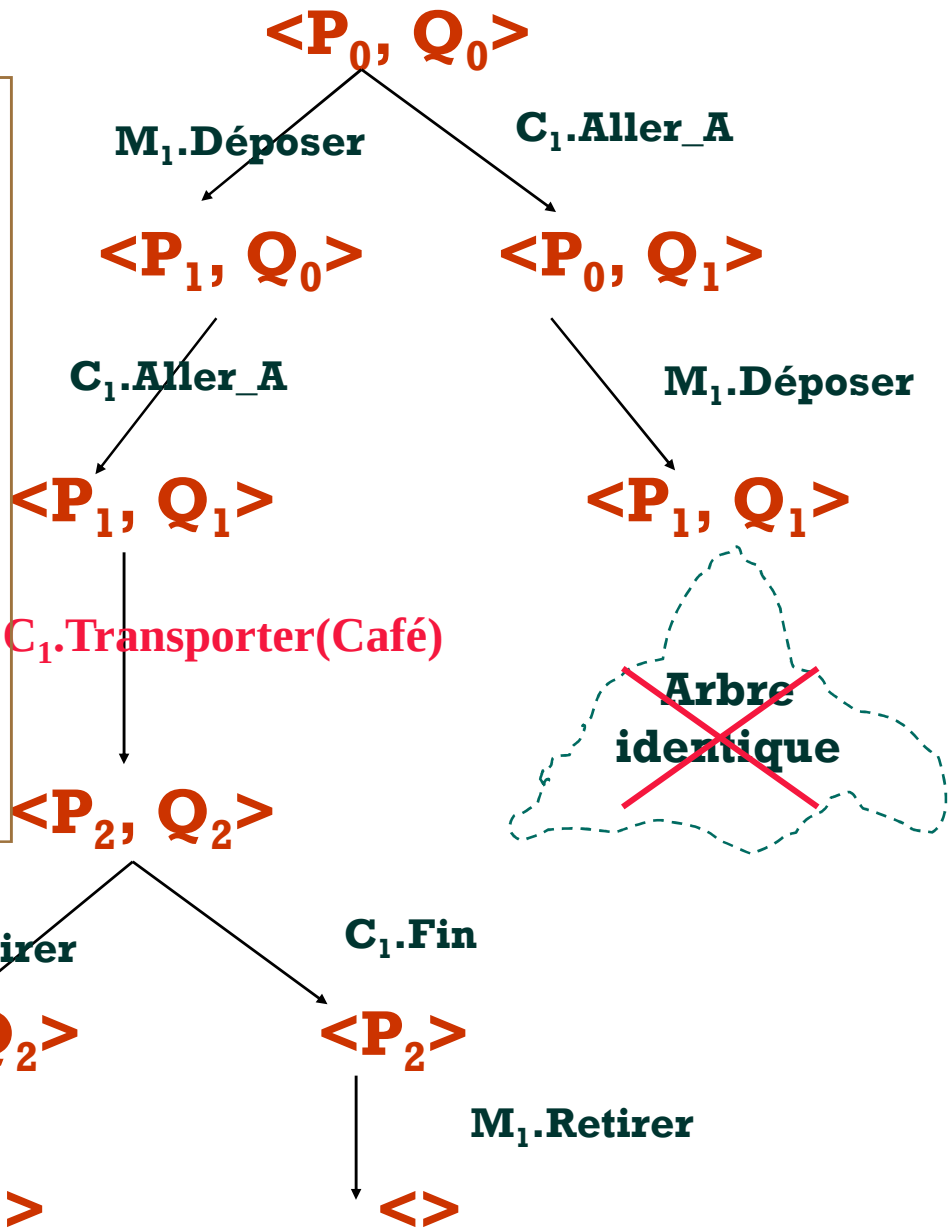
- Vérification de la consistance du plan

  - Développement du graphe d'accessibilité
    - Génération de toutes les séquences de franchissement possibles (toutes les exécutions potentielles du plan)
    - Optimisation de l'arbre d'accessibilité par des techniques de réduction

  - Calcul des pré- et post-conditions d'actions
    - Appliqué au graphe d'accessibilité
    - Tient compte du contexte courant d'exécution (via les conditions sémantiques i.e. les Pré- et Post-Conditions d'actions)

Exemple de construction du G.A.





Vérification de la consistance du plan

```
Function Est_Faisable(in  s: node;  in c: context) :  Boolean;
 begin      {contexte représente les valeurs des  paramètres des méthodes}

        for all  arcs (s, s') in A do
            {A est retourné par l'algorithme de construction de l'arbre d'accessibilté}
           if Evaluate((s, s').Trans.Method.Pre, c) then
                {si les Pré-conditions du Contexte courant sont satisfaites}
              if not ( Est_Faisable(s, Apply((s, s').Trans.Method.Post, c))) then
              return(false)
                                {il existe un sous-arbre non faisable
              endif                  i.e. the post-conditions non satisfaites}
            else return(false)
           endif

        endfor
 return(true)
end {Est_Faisable}.
```

Traitement des interactions négatives

Cas de conflit de ressources
impliquant un convoyeur et deux marchands

Traitement des interactions négatives

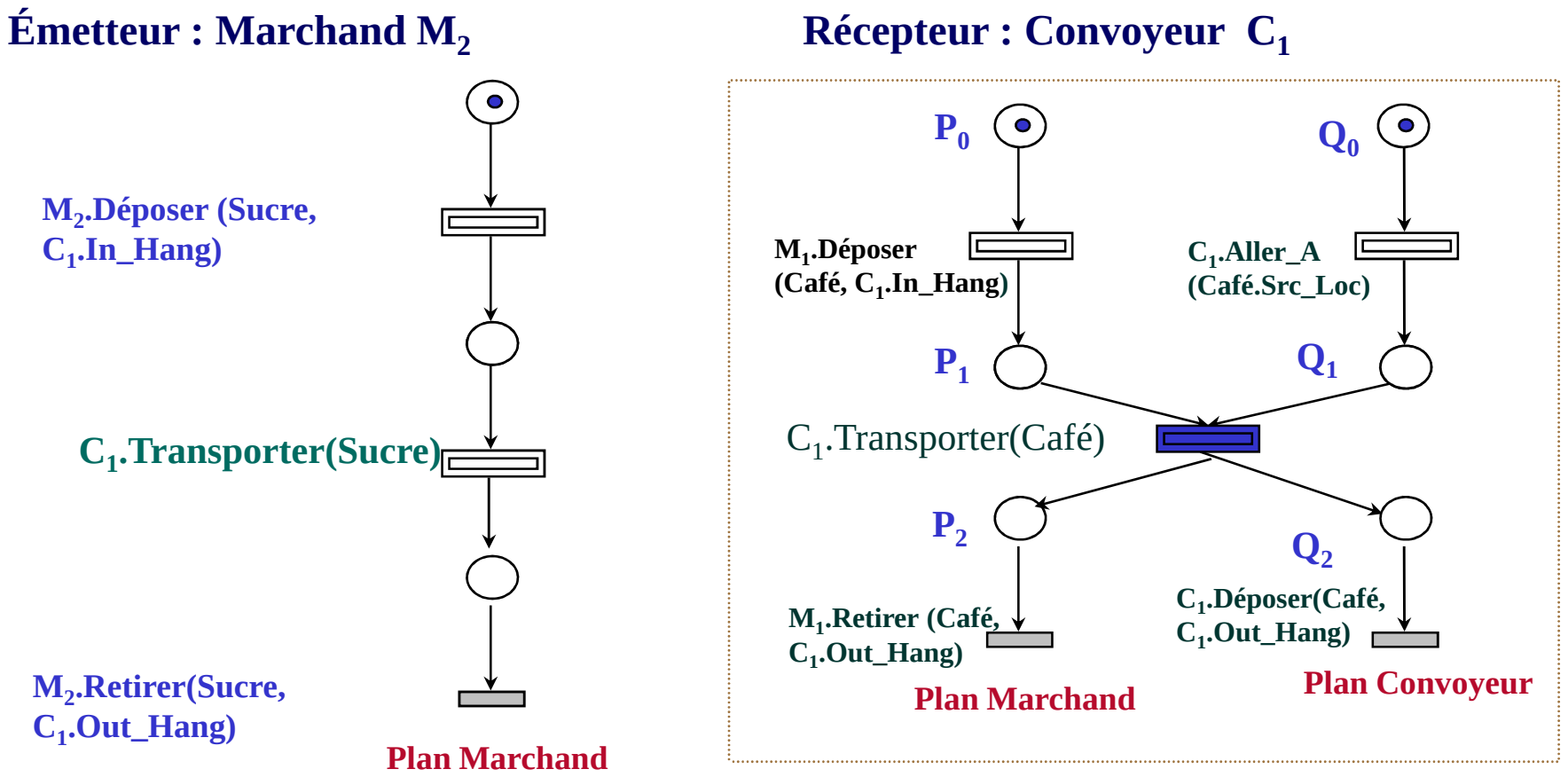
- Contexte :
  - Agent en train d'exécuter une instance de plan et reçoit une nouvelle requête concernant le même raffinement
  - *Interaction négative si les deux raffinements partagent des objets (ou attributs d'objets) source de conflit*

- Génération du second raffinement

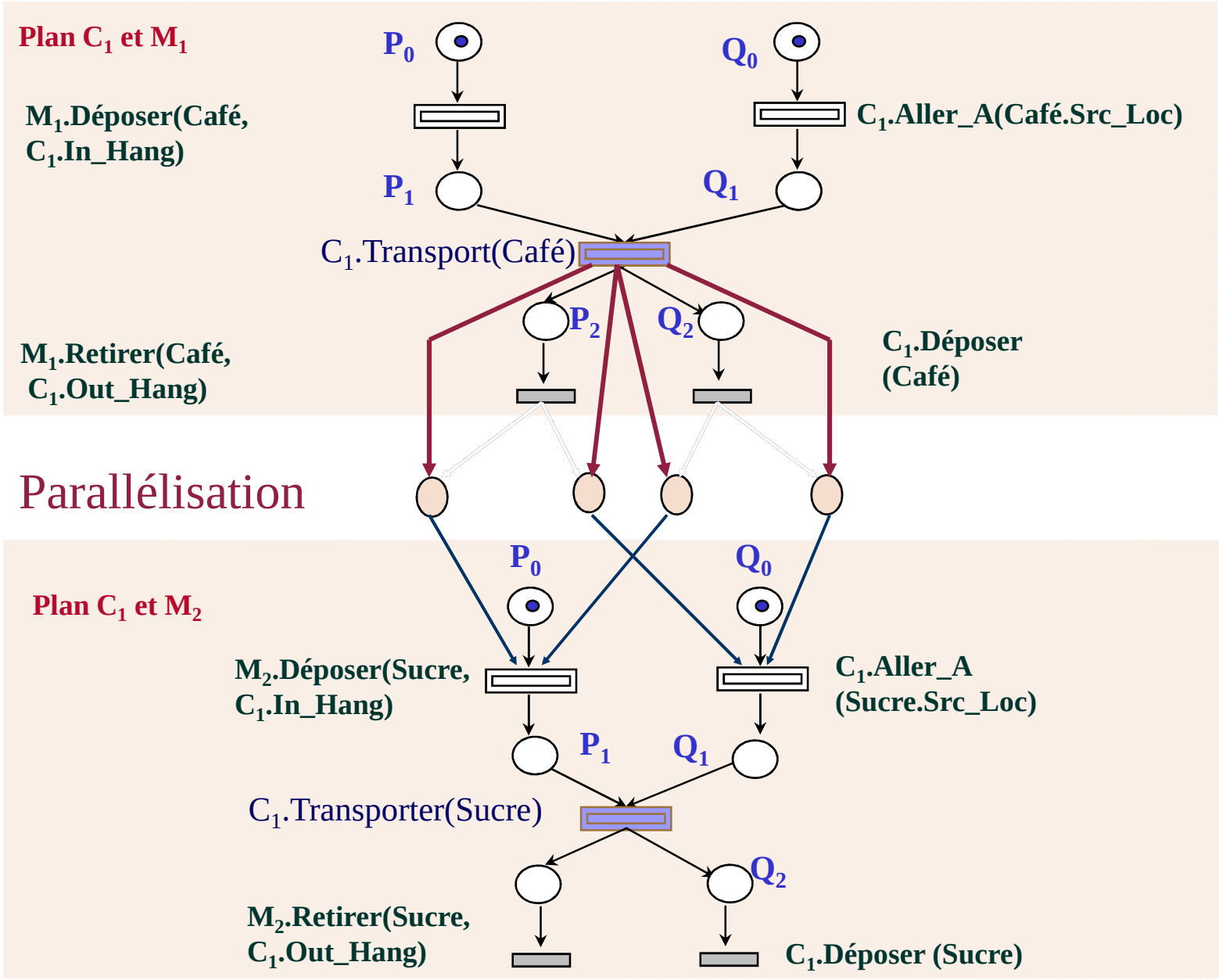
  même principe que le cas d'interaction positive

- Test de conflit par rapport aux objets

Fusion pour traitement d'interaction négative



Séquentialisation-Parallélisation



Discussion du modèle et quelques problèmes de mise en oeuvre (implémentabilité)

Problème lié au raffinement (1)

- Le raffinement impose des contraintes sur les plans raffinant les transitions abstraites

- Notion d'héritage
  - objectif : s'assurer qu'il y a isomorphisme entre les post-conditions d'une transition abstraite ta et celles de tout plan la raffinant
    - indispensable au calcul de consistance

Problème lié au raffinement (2)

- Condition
  - Dans tout ordre total qui étend le RdPR, les post-conditions initiales apparaissent une fois et une seule et dans le même ordre que la transition abstraite.
- Sous cette condition, nous montrons que
  - Si l'algorithme PPCC retourne vrai, alors le plan distribué est faisable.

Dépendances entre agents

- Dépendance de savoir-faire
  - Un agent est « incapable » d'exécuter une action
  - Gestion assurée par les mécanismes précédents

- Dépendance de ressources
  - Un agent « sait faire » mais manque de ressources
  - Nécessite l'accès aux ressources des autres.

Gestion distribuée de ressources partagées

- Objectifs
  - Assurer la cohérence d'accès concurrents aux ressources partagées
  - Garantir l'équité d'attribution des ressources
- Hypothèses : chaque agent
  - dispose de ressources propres (objets locaux)
  - dispose de ressources partagées avec les autres agents regroupées en un pool
  - est responsable de la gestion de son pool de ressources

Ressources partagées

- Variables du plan : paramètres des Pré- et Post-Conditions associées à une méthode
- Exécution d'une méthode : accès en lecture aux paramètres (Pré-) et /ou en écriture (Post-)
- On parle de ressources dans le cas d'objets potentiellement partagés (sinon données)

Principe de gestion de ressources

- Projection de plan sur des pools de ressources
- Détection des dépendances
- Requêtes envoyées aux agents détenant les ressources requises
- Accès basé sur le protocole de diffusion atomique

Phase 1 : Détection d ’une dépendance de ressources

- Un agent A_1 raffine une transition abstraite et :
  - génère son plan Π_1
  - instancie les variables de Π_1
  - constitue V_r : ensemble des ressources requises pour exécuter Π_1
  - partitionne V_r en sous-ensembles d’objets O_j relatifs aux agents A_j

Phase 2-3 : Projection de plan et émission de requêtes

- Phase 2 : projection de plan
  - Pour chaque O_j, A_1 génère une projection $\Pi_{1,j}$
  - Une projection est un sous ensemble de transitions de Π_1 avec leurs places et arcs entrants et sortants et faisant appel à des objets de O_j.

- Phase 3 : Requêtes de demande de ressources
  - A_1 envoie une requête à chaque A_j dont il dépend
  - Une requête est de la forme : (Em1, Rec $_j$, $\Pi_{1,j}$)

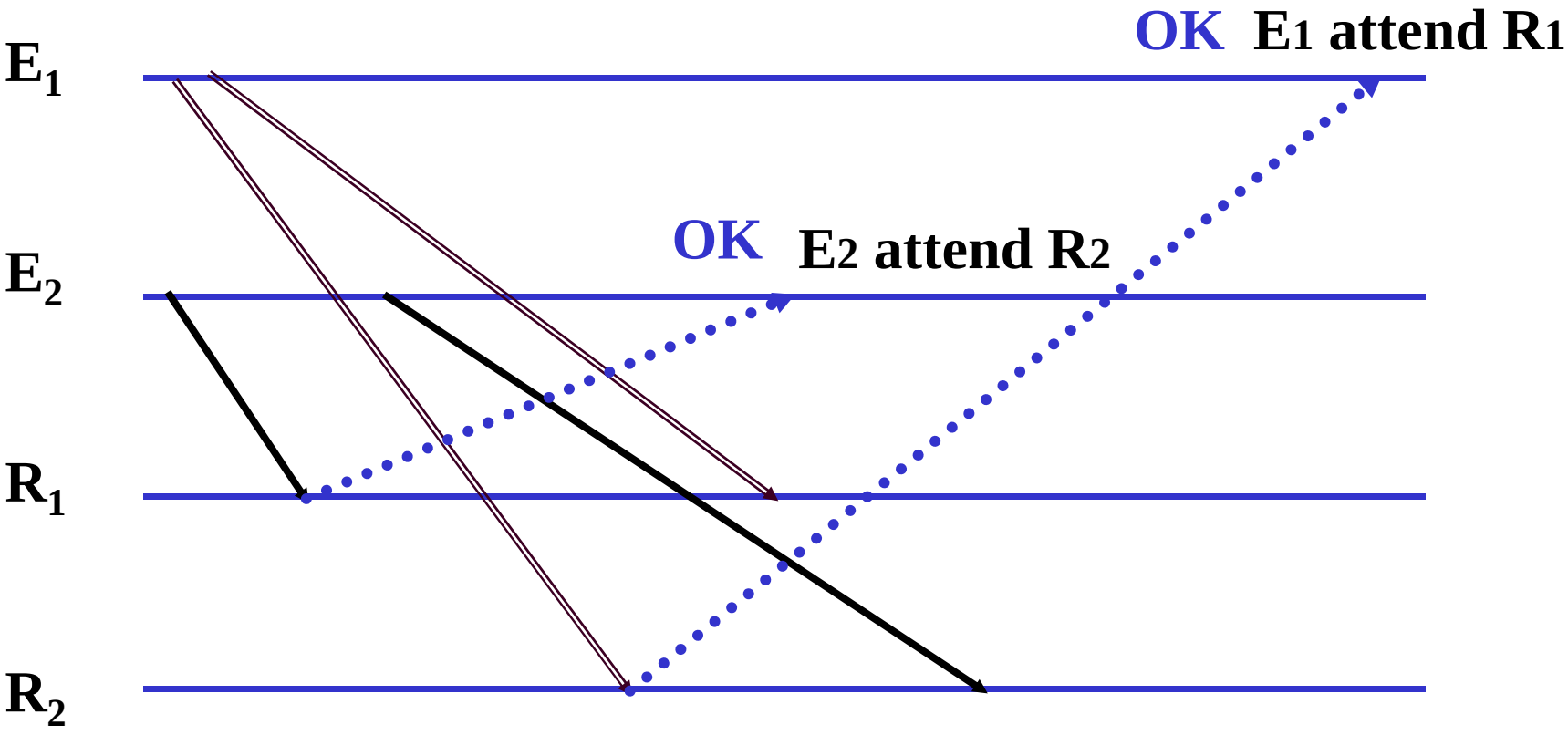
Protocole de diffusion atomique
[Chang et Maxemenchuc, 1984]

- L'agent A1 envoie ses requêtes en respectant le protocole de diffusion atomique et se met en attente des réponses.
- Ce protocole impose que :
  - si A_i envoie des requêtes à A_j et A_k alors, toutes les requêtes envoyées à A_j et A_k arriveront nécessairement dans le même ordre ($j < k$).
- Ce protocole garantit l'absence d'interblocage entre agents demandeurs.

Protocole en 3-Phases

- Etape 1 : diffusion de la requête par l'émetteur (e.g. A_i) dans un certain ordre,
- Etape 2 : chaque récepteur (e.g. A_j et A_k) répond (notification d'acceptation ou de rejet)
- Etape 3 : l'émetteur doit également informer les récepteurs du maintien ou non de son plan ainsi qu'en cas de libération des ressources.

Intérêt des 3-Phases : phase 1 ?



Cas trivial d'interblocage

D’autres cas de disfonctionnement ...

- Phase 2 non achevée
  - les émetteurs en attente infinie
- Phase 3 non achevée
  - un agent reçoit un OK et un KO et n’abandonne pas son plan..
    - La ressource accordée sera indéfiniment détenue
  - un agent ne libère pas les ressources après avoir exécuté son plan …
    - les ressources détenues ne seront jamais restituées

A la réception d'un plan projection

- Le récepteur :
  - séquentialisation/parallélisation de son plan courant avec la projection reçue
  - si plan parallélisé faisable (CCPP)
    - il accepte la requête et retourne la notification OK
    - il renvoie la projection instanciée et augmentée des places de synchronisation et des nouveaux arcs.
  - Sinon
    - renvoie une notification de rejet

- A la réception de plusieurs projections
  - si les projections ont des intersections non vides
    - si demande d'accès en lecture => aucune gestion particulière (simple instanciation des variables)
    - si demande en lecture/écriture => à nouveau séquentialisation/parallélisation, etc.

Extensions du modèle récursif

- Étude d'aspects théoriques des RdPR
  - Résultats sur la décidabilité du problème d'accessibilité et d'autres propriétés des RdPR (S. Haddad et D. Poitrenaud)

- Développement du prototype « RAPID » dans le cadre d'une CTI-CNET « *Surveillance et diagnostic du réseau téléphonique* ».

Conclusion

- Modèle de planification proposé
  - repose sur la planification distribuée et la gestion distribuée des ressources unifiées à travers un même modèle
  - permet de caractériser finement différentes situations d'interaction positives et négatives
  - traite dynamiquement ces situations tout en garantissant des accès concurrents aux ressources qui soient équitables et cohérents
  - offre un cadre formel pour la planification distribuée qui peut être interprété et contrôlé.
  - en tant que modèle RdP théorique : une extension stricte qui préserve les bonnes propriétés.
  - a été implanté : prototype RAPID dans le cadre d'une CTI-CNET
  - a forcément des limites ... *e.g.* suppose les communications fiables.

Quelques Références

- Planification Distribuée
  - Algorithme de coordination de *n* agents (C.O.A.)
    - MAAMAW'96
  - Modèle théorique des RdPR
    - ICMAS '96
  - Gestion distribuée des ressources
    - MAAMAW'97
  - Aspects implémentation et application
    - RIA : numéro spécial sur les SMA. Janvier 98.

Conclusion

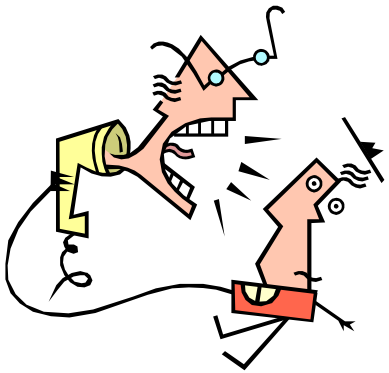
- Modèle de planification proposé
  - repose sur la planification distribuée et la gestion distribuée des ressources unifiées à travers un même modèle
  - permet de caractériser finement différentes situations d'interaction positives et négatives et de les traiter dynamiquement
  - offre un cadre formel pour la planification distribuée qui peut être interprété et contrôlé.
  - en tant que modèle RdP théorique : une extension stricte qui préserve les bonnes propriétés.
  - a été implanté : prototype RAPID dans le cadre d'une CTI-CNET
  - a été étendu pour une gestion distribuée des ressources garantissant des accès concurrents équitables et cohérents

Plan

1. Problématique de la coordination
2. Approches de coordination
3. Coordination par planification distribuée
4. Coordination par formation de coalitions
5. Conclusion

Coordination par formation de coalitions

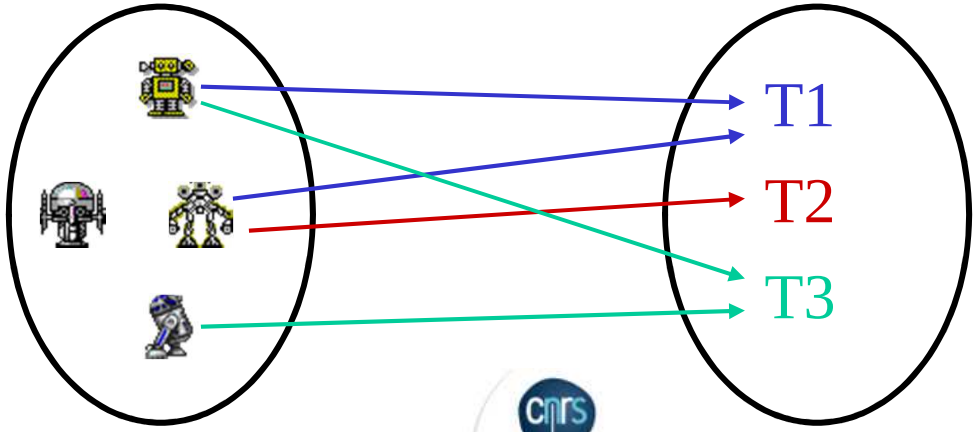
Thèse de Guillaume Vauvert



Exemple de répartition de tâches

- Union de compagnies aériennes -

- Le système reçoit un ensemble de tâches
  - ensemble de vols
- Tâches divisées en sous-tâches
  - Un vol : suite d'étapes
  - Une sous-tâche : compétences + gain



Application : compagnies aériennes

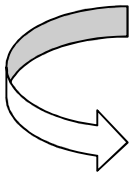
- Objectifs
  - Système unifié de réservation
  - Répartition satisfaisante des étapes
  - Gestion de la concurrence et de la rationalité économique
- Exemple
  - Agents
    - {EUAL, AMAL, WOAL, USAL, AFAL, FRAL, BUAL}
  - Tâches
    - Sous-tâches = {New York - Madrid (via Paris et Lyon), Los Angeles - Moscou (via New York et Paris), Berlin - Johannesburg (via Paris)}
    - + gains associés
  - Compétences
    - {autorisation de survol d'un pays, capacité en passagers, rayon d'action}

Contexte

- Agents du SMA
  - Cognitifs et spécialisés (compétences spécifiques et limitées)
  - Compétitifs (contexte économique)
  - Hétérogènes (stratégies variables)
- Tâches du SMA
  - Tâches décomposables en sous-tâches
  - Réalisation d'une tâche requiert plus d'un agent
- But du SMA
  - Atteinte de consensus pour la répartition des tâches

Comportement des agents cognitifs

- Chaque agent possède
  - Une rationalité économique
    - Implémentée sous forme de stratégies
    - Une stratégie se traduit par des préférences variables
  - Un ensemble de compétences
    - Liées aux sous-tâches
    - Ne couvrent pas entièrement une tâche



Le comportement d'un agent résulte de sa rationalité et de ses compétences

Formation de coalitions

- Coalition
  - Organisation dynamique
    - Environnements ouverts et dynamiques
  - Engagements ponctuels et contextuel
    - Réactions opportunistes et dynamiques
  - Formation / Dissolution
    - Contexte-dépendante

- Coalitions appliquées à la répartition de tâches
  - Organisation pour coordonner les comportements des agents
  - Coalition : ensemble d'agents coopérant pour résoudre une tâche divisée en sous-tâches
    - Synergie des compétences
    - Groupement par intérêt

Problème de formation de coalitions

- Un PFC est défini comme un 5-uplet <A, T, S, C, P> où :
  - A : Agents candidats pour exécuter des sous-tâches
  - T : tâches à accomplir
  - S : Sous-tâches à accomplir
  - C : Compétences
  - P : Profit (gain associé à une sous-tâche)
- Solution: affectation de toutes les sous-tâches
  - application $\sigma : S \rightarrow A$, prenant en compte les compétences
- But : atteinte de consensus concerant une solution

Quel protocole de coordination ?

- Hétérogénéité : Problème des interactions
  - Soit standardisation (à la FIPA) : permet argumentation
  - Soit simplification (échange de préférences)
- Rationalité économique
  - Incitation à la coopération (bénéfique pour les agents)
  - Prévoir de contraindre si nécessaire
  - Indépendant des stratégies (universel)
  - Ne pas favoriser d'agent (égalitaire)
- Forte autonomie
  - Prévenir la fraude (vérification + sanction)
  - Comportement éventuellement complexe (rationalité limitée, calcul sur le long / court terme …)
  - Aucune hypothèse sur le comportement des agents

Protocole de formation de coalitions

- Objectif : Parvenir à une répartition des sous-tâches entre les agents par atteinte d'un consensus

- Algorithme distribué basé sur la négociation
  - Échange de préférences
  - Contrôle du respect du protocole
  - Garantie de l'impartialité
  - Favorisation de l'atteinte d'un consensus par la formation d'alliance
  - Etablissement d'une règle qui garantit l'atteinte d'un consensus

Principes du protocole

- Calcul des préférences
- Echange de préférences
- Formation d'alliances en cas de blocage

Représentation des préférences

- Préférences :
  - « distances » entre solutions
  - Pas de messages complexes (hétérogénéité)
  - Exemple :

    X préfère largement σ_1 à σ_2 $\Leftrightarrow \delta(\sigma_1, \sigma_2)=.9$

- Matrice antisymetrique :

	σ_1	σ_2	σ_3	σ_4
σ_1	0	.9	-.2	0
σ_2	-.9	0	-1	.1
σ_3	.2	1	0	-.5
σ_4	0	-.1	.5	0

Calcul des préférences

- Préférences initiales

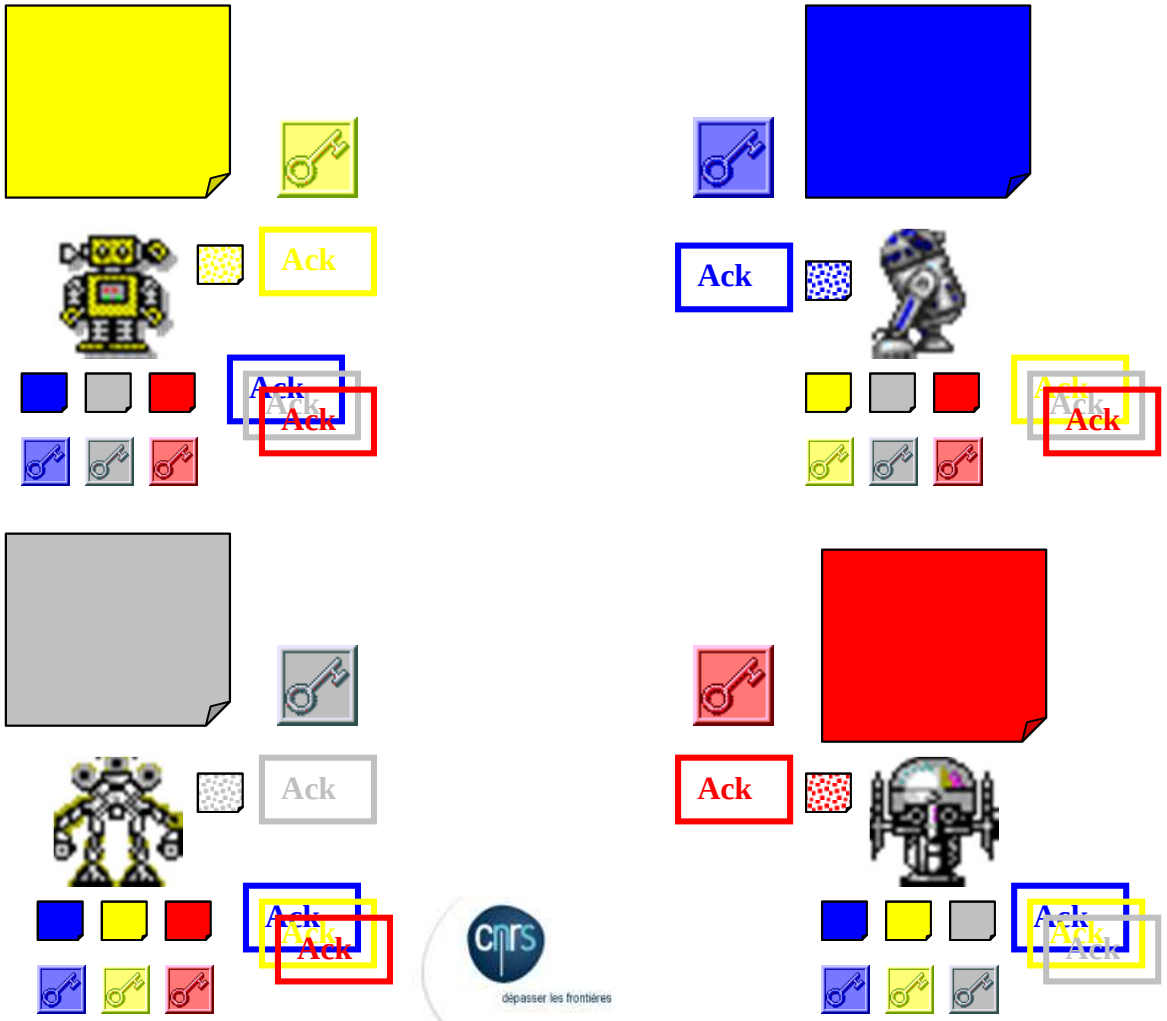
$$\delta_a^0(\sigma_1,\sigma_2)=\frac{income(\sigma_1)-income(\sigma_2)}{Sup_\sigma income(\sigma)}$$

- Préférences dépendantes

$$\delta'_a(\sigma_1,\sigma_2)=\omega\times\delta_a^0(\sigma_1,\sigma_2)+(1-\omega)\frac{\sum_{b\in A\setminus\{a\}}\delta_b(\sigma_1,\sigma_2)}{|A|-1}$$

  - Fléxible: $\omega \to 0$; Rigide: $\omega \to 1$

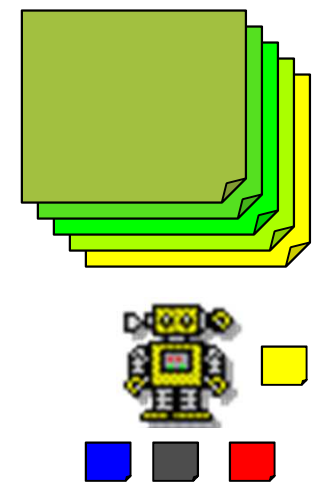
Diffusion parallèle

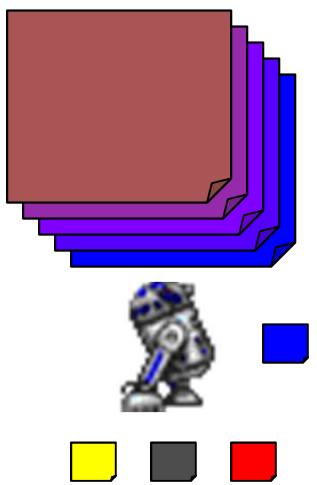


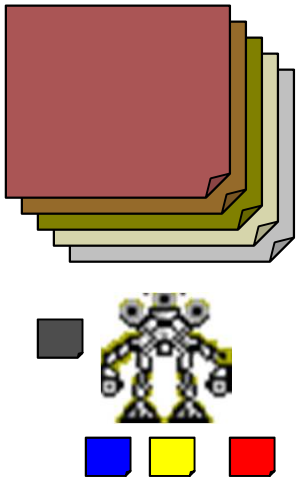
Algorithme - Diffusion parallèle

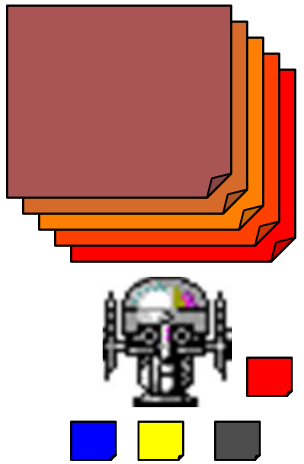
- For all $\alpha \in A$ do
  - $\theta* \leftarrow$ Encrypt(θ, key)
  - broadcast($\theta*$)
    - *{ diffuser l'information cryptée }*
  - receiptAll($\theta*$, $A \backslash \alpha$)
    - *{ attendre toutes les réceptions }*
  - broadcast(« Ack »)
  - receiptAll(« Ack », $A \backslash \alpha$)
  - broadcast(key, $A \backslash \alpha$)
    - *{ diffuser la clé }*

Echange de préférences



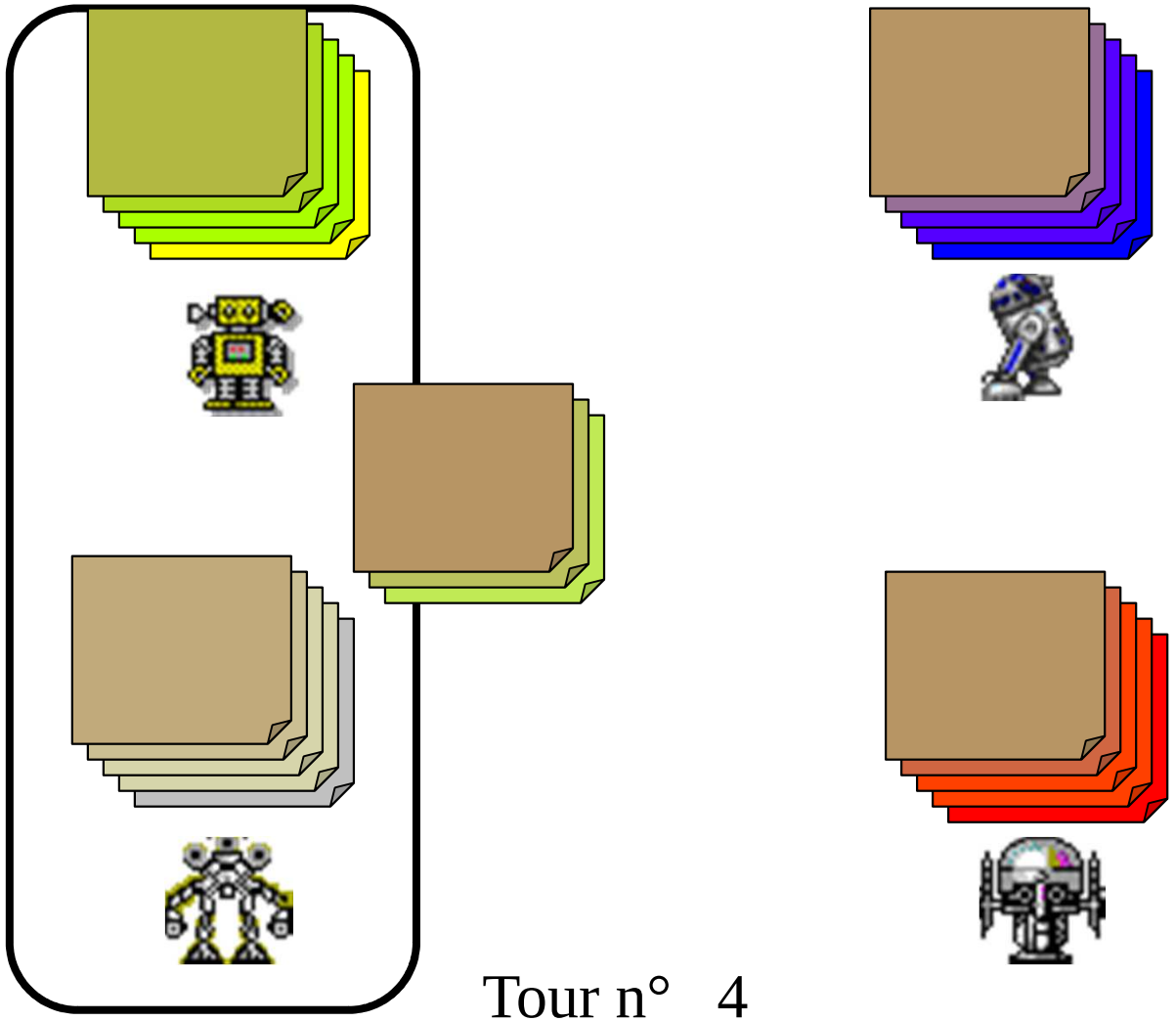






Tour n° 4

Formation d’alliances



Tour n° 4

Algorithme - Les préférences

- IndPref ← IPC *{Calcul des préférences initiales}*
- h ↵ DiffParallèle(IndPref) *{Diffusion et mise à jour}*
- While (¬ consensus) do
  - If RSPC(h) Then sendAll(«Mode déblocage ?»)
  - If receive(«Mode déblocage ?») Then
    - If RSAC Then sendAll(«accepte déblocage»)
  - If receiveAll(«accepte déblocage»)
    - Then call mode_déblocage
  - DepPref ← IPC *{Calcul des préférences dépendantes}*
  - h ↵ DiffParallèle(DepPref)

Algorithme - Mode déblocage

- Broadcast(«propose formation alliance», AFPC(h))
  - *{ AFPC : ensemble des agents à qui proposer une alliance }*
- For each a / receive(«propose alliance»,α) do
  - If $\alpha \in$ AFAC(h) Then send(«j'accepte», α)

    { AFAC : décide si la proposition de α est acceptable }
  - If (pas d'alliance formée) Then
    - $B \leftarrow$SMA.nearestAgents()
    - *{SMA.nearestAgents() : fonction commune connue au début du processus }*
  - Les membres de B doivent former une alliance

Terminaison

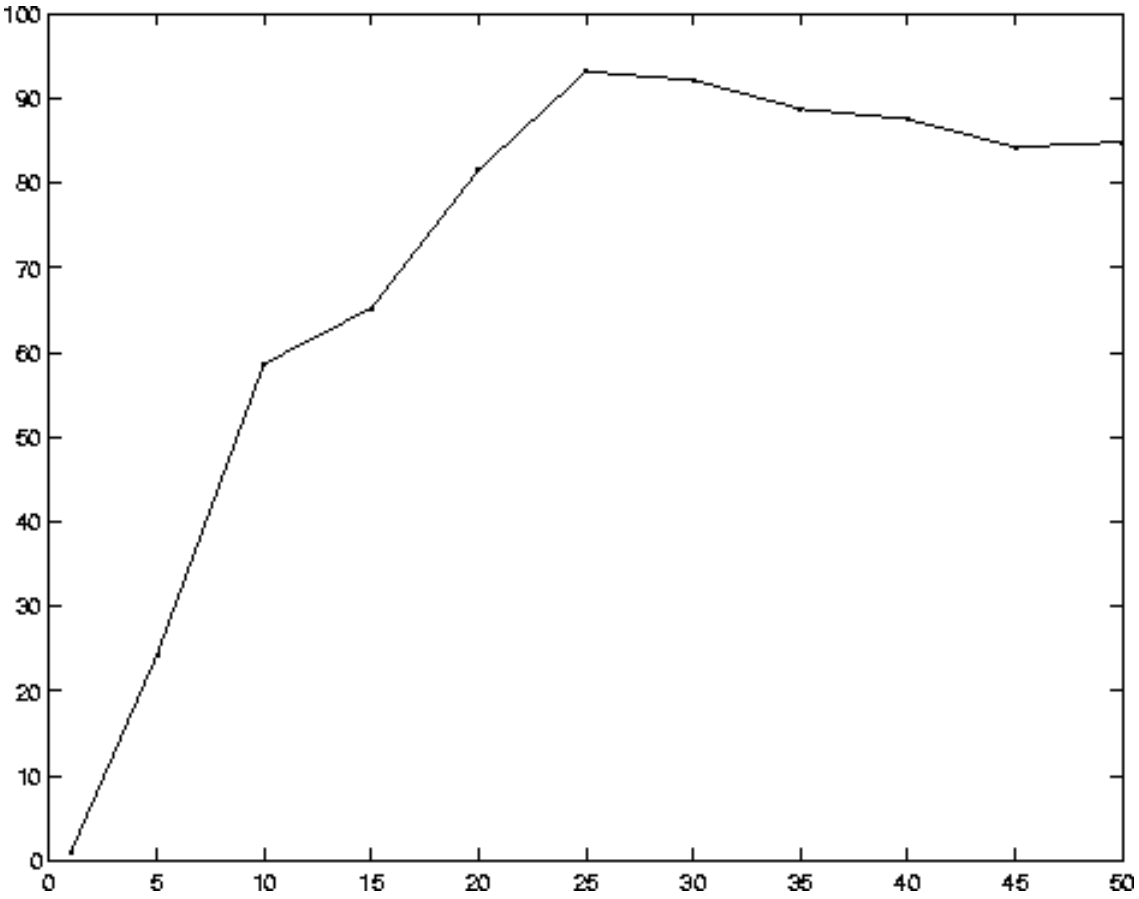
- Définition : une histoire contient une boucle si une situation se présente deux fois.
- Théorème : si un CFP détecte les boucles, alors le processus termine.
- Preuve :
  - nombre fini de solutions
  - consensus si tout le monde a la même préférence

Résultats de la simulation

- Nombre d'agents
  - Incidence sur la vitesse de convergence
- Stratégie
  - Incidence sur les gains des agents
  - Incidence sur la vitesse de convergence
- Compétition
  - Incidence sur les gains des agents
  - Incidence sur la vitesse de convergence

Résultats

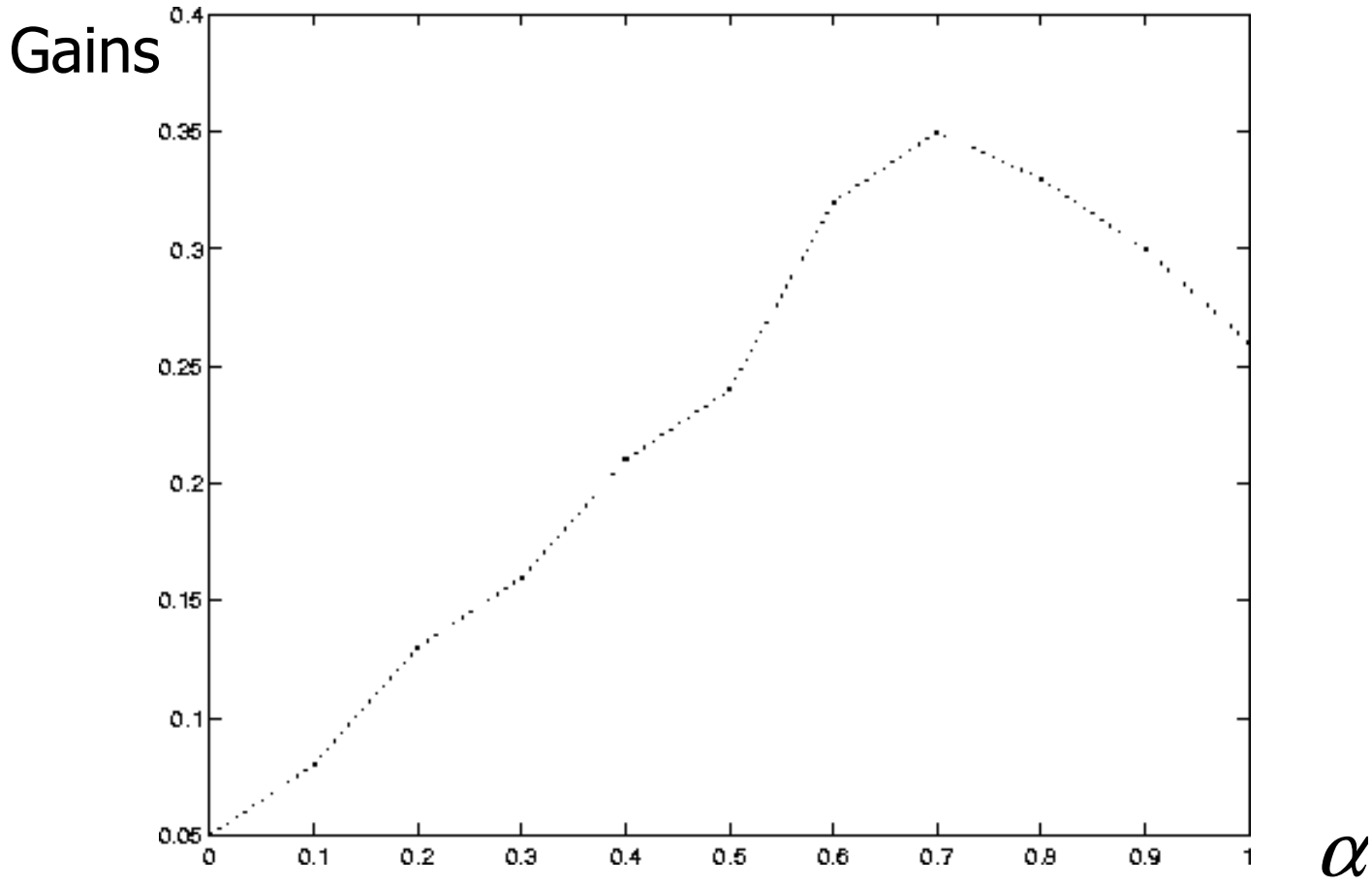
Nombre de tours



Nombre d'agents

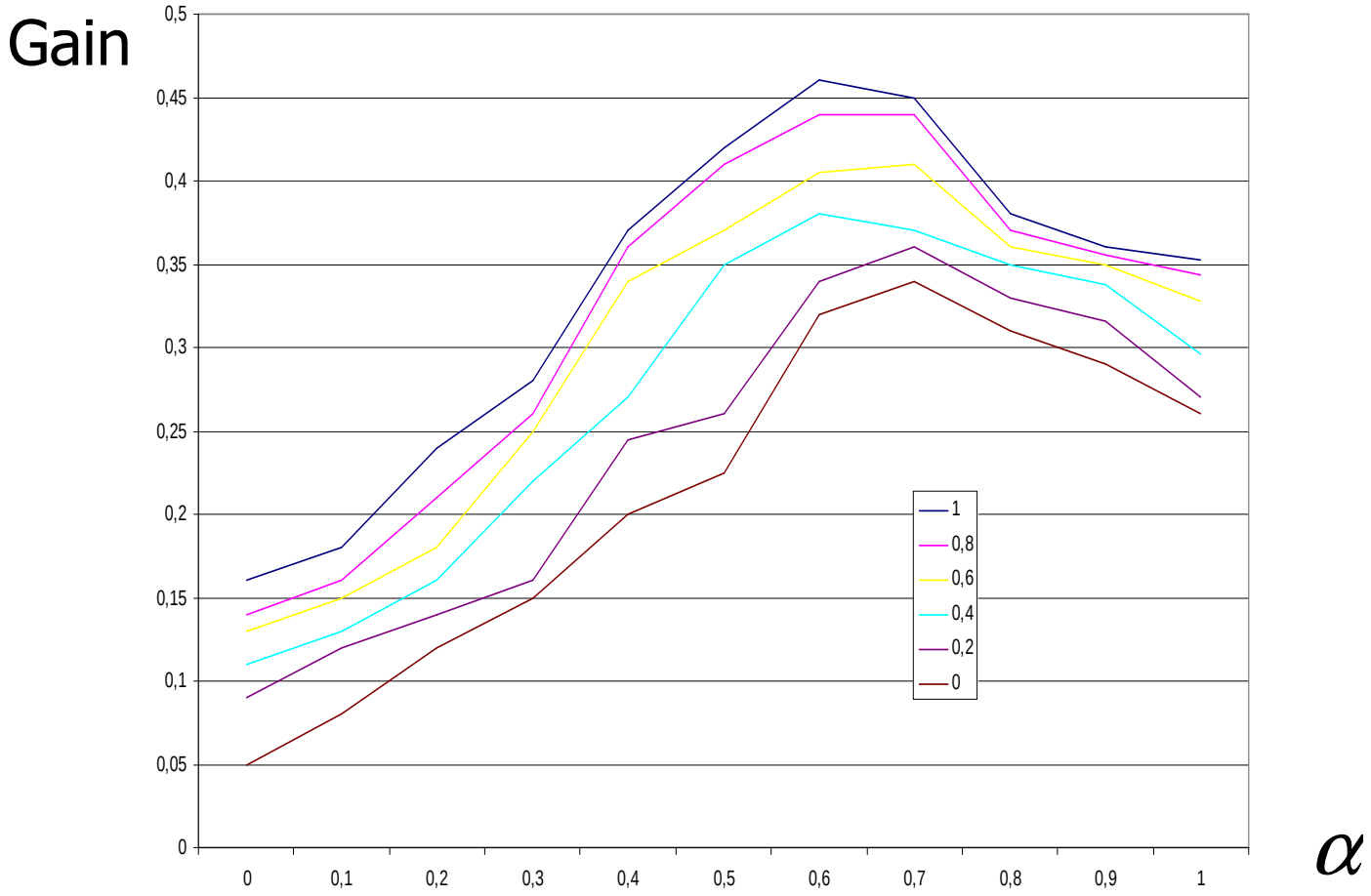
Nombre de tours / Nombre d'agents

Résultats



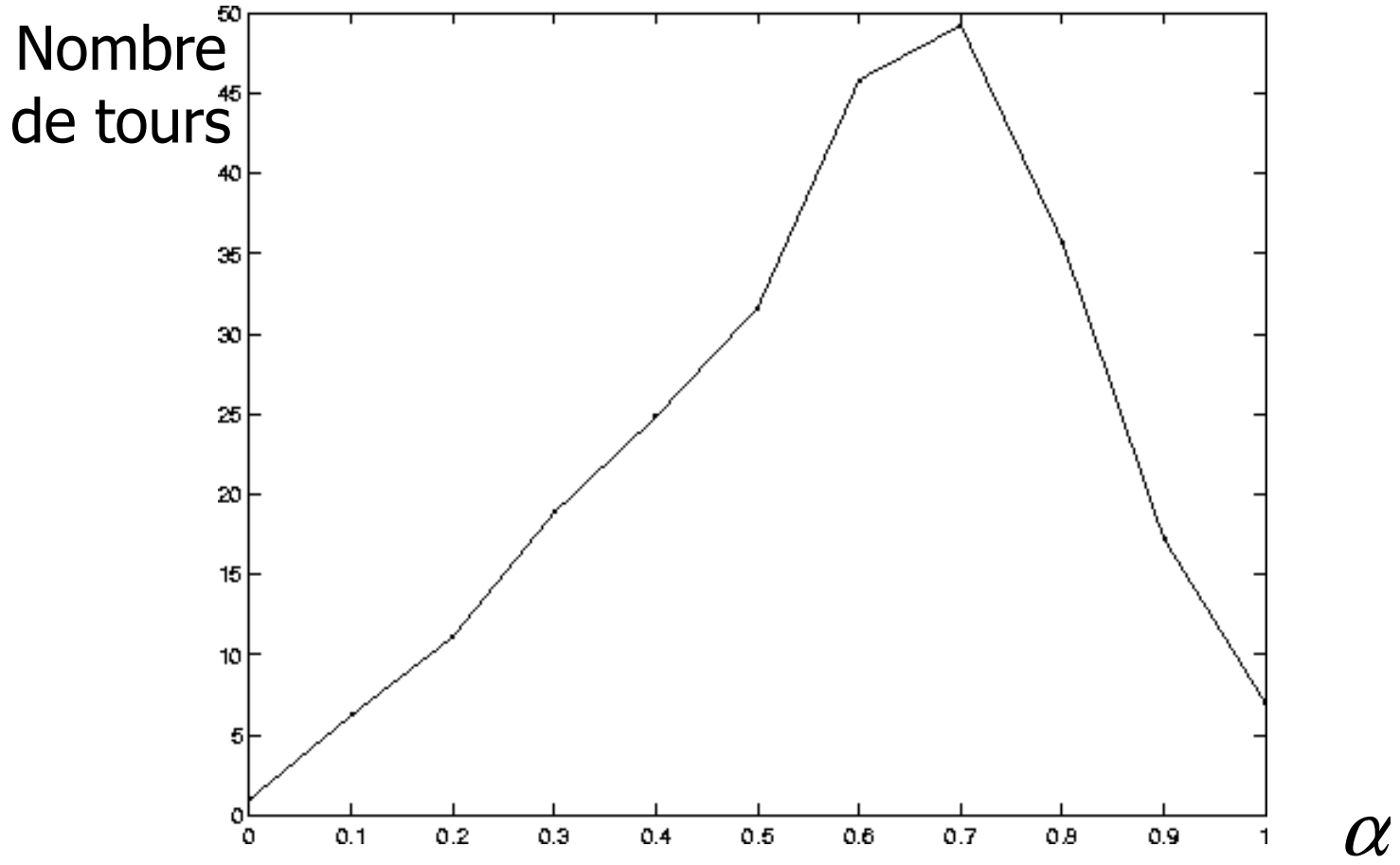
Gain d'un agent face à une population uniforme

Résultats



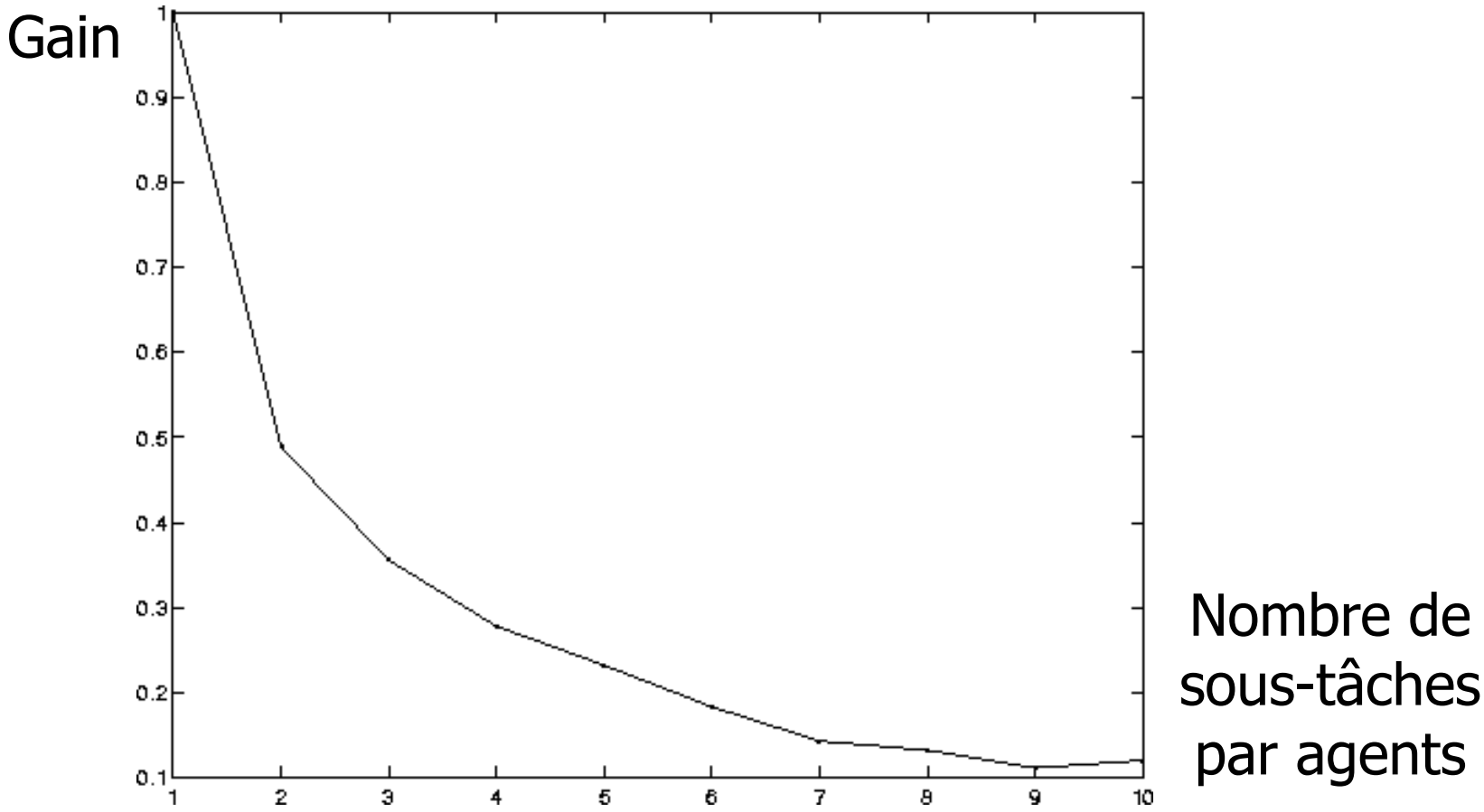
Gain d'un agent face à une population uniforme

Résultats



Vitesse de convergence dans le cas d'une population uniforme

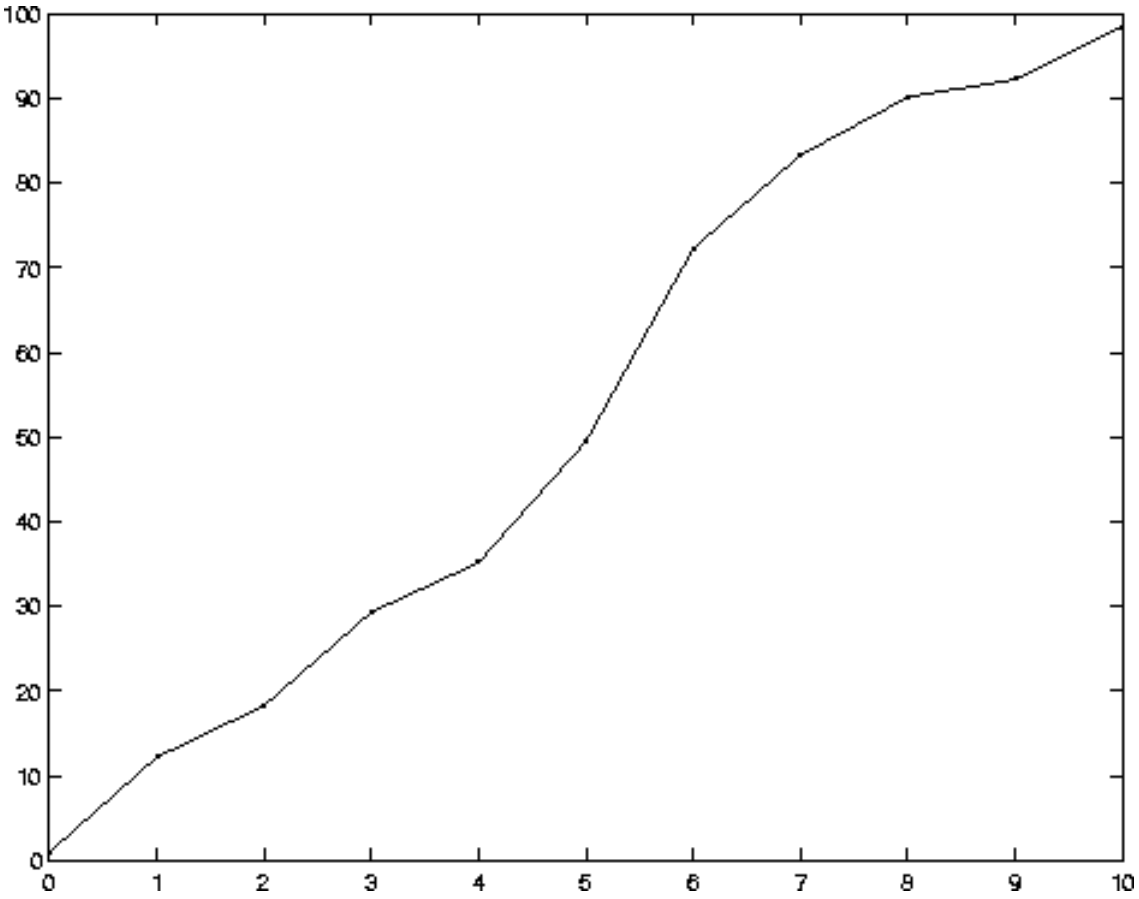
Résultats



Gain d'un agent / Nombre de sous-tâches par agent

Résultats

Nombre de tours



Nombre de sous-tâches par agent

Nombre de tours / Nombre de sous-tâches par agent

Conclusion

- Quel Modèle de coordination pour quel SMA ?
  - Systèmes orientés tâches
  - Systèmes orientés agents
    - Agents coopératifs
    - Agents compétitifs
- Critères d'évaluation
  - Efficacité,
  - Qualité de la solution
  - Tolérance aux pannes
  - Adaptabilité
  - Réactivité