Multi Agent System

CNP

Il Contract Net Protocol è un protocollo di interazione per l'allocazione dei task.

Esso è utilizzato da un agente, chiamato **INIZIATOR**, che desidera avere un task svolto da uno o più altri agenti, chiamati **PARTECIPANTI**.

Il flusso del protocollo è il seguente: **L'INIZIATOR** chiede agli altri agenti di presentare delle proposte (Call For Proposal) per specifici task. I partecipanti ricevono le CFP e possono presentare le loro proposte di svolgere un task che comprendono precondizioni della loro esecuzione del compito, come il costo, come il momento in cui sarà effettuata l'operazione o qualsiasi altra precondizione. In alternativa, gli agenti possono rifiutare. Non appena i partecipanti finiscono le loro valutazioni, l'iniziatore valuta le proposte ricevute e seleziona gli agenti per eseguire l'operazione. Si può scegliere uno, alcuni o nessuno se non ci sono proposte soddisfacenti. All'agente che ha fatto la proposta migliore verrà inviata la notifica di accettazione, e gli agenti rimanenti riceveranno rifiuti. Non appena l'iniziatore accetta la proposta, il partecipante ha un impegno a svolgere il compito. Dopo aver completato l'operazione il partecipante invia il messaggio di completamento all'iniziatore. In caso di guasto, un messaggio corrispondente viene inviato. Inoltre, i messaggi non-capiti possono essere inviati in qualsiasi momento del protocollo nel caso in cui l'agente non ha capito il messaggio precedente. In generale, il Contract Net Protocol consente di trovare l'agente più adatto per una singola attività confrontando proposte presentate dagli agenti. Utilizzando questo protocollo, si hanno tali vantaggi:

1. Le attività vengono assegnate dinamicamente.
2. Gli agenti possono entrare e uscire dal sistema a proprio piacimento.
3. I compiti saranno naturalmente bilanciati tra tutti gli agenti in quanto gli agenti che hanno già uno o più task assegnati non faranno un'offerta per quelli nuovi. Se un agente sta già usando tutte le sue risorse, non sarà in grado di fare un'offerta per nuovi contratti finché i task attuali non sono stati completati.
4. Una strategia affidabile per applicazioni distribuite con gli agenti che possono recuperare dai fallimenti.

Purtroppo, Contract Net Protocol ha anche una serie di carenze:

1. Non vi è alcun meccanismo per rilevare conflitti e soprattutto risolverli.
2. L'infrastruttura di comunicazione non è completamente affidabile. Il sistema deve avere meccanismi per cercare di reindirizzare comunicazioni in caso di guasto.

SWARM ROBOTICS

Swarm robotics è lo studio di come un gran numero di agenti possono essere progettati in modo tale che un comportamento collettivo desiderato, emerge dalle interazioni locali tra agenti e tra gli agenti e l'ambiente.

I robot dello swarm devono essere robot autonomi, in grado di percepire ed agire in un ambiente reale. I robot hanno solo capacità di comunicazione e di rilevamento locale. Ciò assicura il coordinamento distribuito, quindi la scalabilità diventa una delle proprietà del sistema. Inoltre i robot devono essere incapaci o inefficienti rispetto al compito principale che devono risolvere e quindi devono collaborare per migliorare le prestazioni.

Possibili comportamenti per Swarm Robotics:

1. ***Aggregazione***: il processo che porta un gruppo di agenti a raggrupparsi in una posizione specifica. Ciò può avvenire in 2 modi:
    1. Ogni agente emette un segnale, il segnale si somma a quello degli altri diventando più attraente. Un feedback positivo conduce verso la formazione di un singolo cluster.
    2. Ogni agente si muove casualmente e si ferma quando ne incontra un altro. Il tempo di fine dipende dalla dimensione del cluster.

2. ***Movimento Coordinato***: il processo che porta un gruppo di agenti di muoversi in modo coerente e ordinato. Inoltre vi è una variante di questo processo in cui un singolo agente ha informazioni sufficienti per condurre l'intero gruppo. **Approccio centralizzato.** Nessun agente è più informato degli altri. **Approccio auto-organizzato.**
    1. Ci sono tre semplici regole locali: l'aggregazione, la repulsione e l'allineamento. Il movimento viene eseguito guardando la posizione e l'orientamento dei vicini o ..
    2. I robot sono fisicamente collegati e percepiscono il movimento degli altri.

3. ***Esplorazione collettiva e Area Coverage***: il processo che porta un gruppo di agenti a disperdersi nell'ambiente alla ricerca di risorse. Le varianti sono la presenza o meno di un luogo di riferimento, se la loro esplorazione avviene in una zona di ricerca aperta o chiusa e la presenza o meno di ostacoli.
    1. Una possibile implementazione: esplorazioni casuali
    2. Creazione di una rete collegata di agenti che si espandono a partire dalla posizione della propria abitazione; l'obiettivo è di ottenere la massima copertura intorno alla posizione domestica creando una struttura navigabile.
    3. Creazione di una catena che si estende dal posto centrale. Questo porta alla massimizzazione della distanza di ricerca, creando sempre una struttura navigabile.

4. P***rocesso decisionale collettivo***: il processo che porta un gruppo di agenti ad identificare la migliore opzione tra le diverse alternative. Le varianti sono la semplice propagazione delle informazioni, la media dei pareri e l'amplificazione delle scelte migliori.
    1. La prima possibile implementazione è fatta dalla scelta tra due alternative, dove l'aggregazione dipende dal numero di individui.
    2. La seconda implementazione è la selezione di un "nido" come con le api: un gruppo di agenti ha bisogno di selezionare un nuovo nido; agenti esploratori identificano le possibili alternative e condividono le informazioni.

Swarm robots eseguono i task avendo come obiettivo quello di coprire una vasta regione di spazio. Il vantaggio principale su una rete di sensori è che il gruppo può muoversi e concentrarsi sul problema e anche agire per prevenire le conseguenze di tale problema. In questo modo, gruppi di robot possono essere molto utili per le attività pericolose, come l'esplorazioni di case distrutte dai terremoti e ancora pericolanti.

DCOP

Distributed constraint optimization (DCOP) ha come obiettivo, non quello di trovare una soluzione, ma la migliore, che richiede quindi più l'esplorazione dello spazio di ricerca. L'obiettivo comune di tutti gli algoritmi distribuiti è quello di minimizzare il numero di messaggi necessari per trovare una soluzione.

Esso è rappresentato da una tupla, < X,D,C >, dove:

$X = \{X_1,\ldots,X_n\}$ a set of variables (e.g. meetings)

$D = \{D_1,\ldots,D_n\}$ a set of discrete variable domains (e.g. time slots)

$C = \{C_1,\ldots,C_m\}$ a set of constraints (e.g., equality, non overlap,)

$S_i \subseteq X$ Scope of constraint C_i

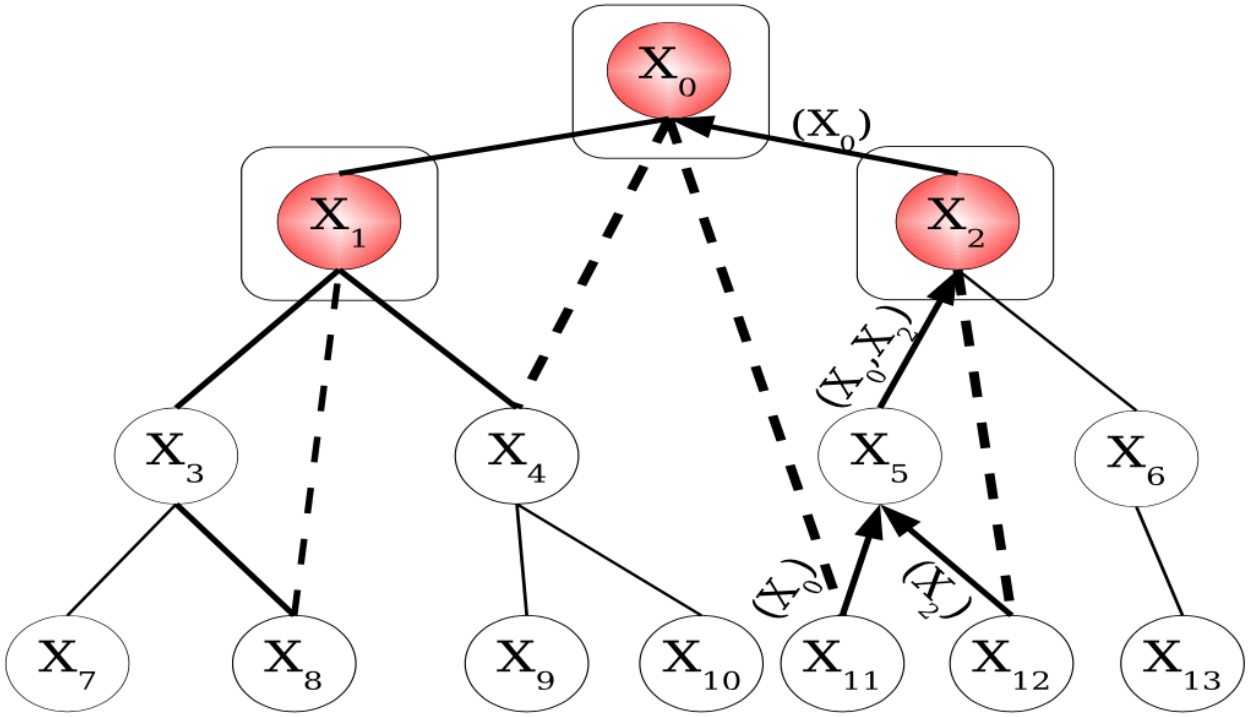
Hard constraints

R_i	x_j	x_k
	0	1
	1	0

Soft constraint

F_i	x_j	x_k
2	0	0
0	0	1
0	1	0
1	1	1

Per le reti con struttura ad albero, è stato sviluppato l'algoritmo DTREE.
Per applicare l'algoritmo DTREE a un grafo ciclico, dobbiamo prima arrangiare il grafo come uno pseudotree.
L'idea principale è che a causa della relativa indipendenza tra i nodi nei diversi rami del pseudotree, è possibile effettuare la ricerca in parallelo su questi rami indipendenti.
Lo pseudotree consiste di tree-edges, indicati come linee continue, e back-edges indicati con linee tratteggiate.
Come sappiamo, un DFS tree è già uno pseudotree ottenuto dal attraversamento DFS del grafo partnendo da uno dei nodi (scelto mediante un algoritmo).
L'algoritmo DPOP consiste di 3 fasi. In primo luogo, gli agenti stabiliscono la struttura pseudotree da utilizzare nelle seguenti due fasi. Le due fasi successive sono UTIL e VALUE propagation, che sono simili a quelli di DTREE.



UTIL PROPAGATION:

Tale propagazione parte dalle foglie del pseudotree e si propaga in alto verso il nodo principale, solo attraverso i tree-edges. È facile per un agente per identificare se si tratta di un nodo foglia o no: deve avere un singolo tree-edge. In una rete ad albero, un messaggio UTIL inviato da un nodo al suo genitore dipende solo dal suo sottoalbero, e dal vincolo tra il nodo e il suo genitore. Ad esempio, si consideri il messaggio (X6 → X2). Questo messaggio è chiaramente dipende solo dalla variabile target X2, dato che non esistono collegamenti tra X6 o X13 e qualsiasi nodo sopra X2.

In una rete con cicli (ogni back-edge nel pseudotree produce un ciclo), un messaggio inviato da un nodo al suo genitore può anche dipendere dalle variabili che si trovano sopra al genitore. Ciò accade quando c'è un back-edges che collega il nodo di invio con una variabile. Ad esempio, si consideri il messaggio (X8 → X3). Si vede che l'Ultil message di X8 non dipende solo dal suo genitore X3 (come per X6 → X2). X8 è collegato con X1 attraverso il backedge X8 → X1, X8 deve tener conto di questa dipendenza quando invia il suo messaggio di X3.

Qui è dove l'approccio della programmazione dinamica entra in gioco: X8 calcolerà le utility ottimali e invierà un messaggio con 2 dimensioni (una per la variabile target X3 e una per il back-edge X1) a X3. Questa è la differenza fondamentale tra DTREE e DPOP: i messaggi che viaggiano attraverso la rete in DTREE hanno sempre una sola dimensione, mentre in DPOP, i messaggi hanno dimensioni multiple (uno per la variabile target, e un altra per ogni context variabile).

Value PROPAGATION:

La fase di Value propagation inversamente a quella dell'Util si propaga dal nodo root ai nodi foglia.

Tale messaggio conterrà il valore sia del tree-edge che del back-edge con tutte le dipendenze.

Ad esempio X_0 sends X_2 V ALU E_0^2 ($X_0 \leftarrow v_0^*$), then X_2 sendsX_5 VALUE$_2^5$($X_0 \leftarrow v_0^*$,$X_2 \leftarrow v_2^*$), andX_5 sends X_{11} VALUE^{11}($X_0 \leftarrow v_0^*$, $X_5 \leftarrow v_5^*$).