Task Assingnment Problem: bibliografia ragionata e prima implementazione

Gruppo 6

31 dicembre 2009

Scopo del presente documento è esporre sintesi della bibliografia esaminata, inerente l'ambito del task assignment problem. Non saranno illustrati solamente i risultati applicabili (o che riteniamo tali) alla soluzione del problema specifico di nostro interesse, ma anche le idee scartate, pur sembrando inizialmente ragionevoli, e quelle che non hanno ancora trovato una formalizzazione oppure un'interpretazione adatta all'applicazione. Nell'ultima parte è presentato l'approccio adottato nella simulazione.

Segue l'elenco degli articoli (e degli approcci da essi suggeriti) finora presi in considerazione.

- **Matthias J. Feiler.** ***On distributed search in an uncertain environment.*** **2009.**

  Basato su tecniche di identificazione dinamica, l'articolo propone una strategia (che inizialmente è sembrata interessante) per assegnare task ad agenti. Dopo aver formalizzato il nostro problema nel dettaglio,
  (`https://docs.google.com/Doc?docid=0AXka0Qh7wGu-ZGN3OTh6NjZfMWR4djI2Z253&hl=en`)
  pur avendo elaborato delle migliorie ed alcune soluzioni ad hoc, alcune considerazioni ci hanno spinto a scartare il metodo:

  1. Numero degli agenti richiesto deve essere superiore a quello dei task: ipotesi incompatibile con la definizione del problema specifico (telecamere sono in numero inferiore ai possibili task)
  2. Convergenza quando si verifica è simultanea per tutti i task: una semplice implementazione ha evidenziato come l'algoritmo fosse lento e ridondante (in effetti non ci serve che ogni task sia assegnato, poichè solo alcuni sono attivi, in realtà. Con l'approccio di Feiler è necessario che tutti si manifestino affinchè ci sia convergenza)
  3. Tendenzialmente, possono manifestarsi deadlocks e comportamenti ciclici (in dipendenza dall'ordine di apparizione dei task)

- **Diego Pizzocaro.** ***Sensor Assignment in a virtual environment using contraint programming.*** **Thesis, 2007**

  La tesi si occupa di un *placement problem*: disporre sensori (Audio, Video, A/V) in un campo di battaglia. Propone un approccio che si articola in due fasi, ispirato ai paradigmi di soluzione del Multiple Knapsack Problem:

  1. Sensor Assignment: massimizzare l'area coperta dai sensori (senza effettuare il posizionamento esatto, con conseguente riduzione del dominio delle variabili decisionali)
  2. Sensor Deployment (con scheduling delle capacità dei sensori nel caso A/V). Applicazione ricorsiva dell'algoritmo MKP.

  Da qui siamo partiti ottenere un modello di programmazione lineare intera per il nostro problema. Abbiamo raccolto ulteriori informazioni sulle varianti dei problemi KP, leggendo tra l'altro l'articolo ***Generation Methods for Multidimensional Knapsack Problems and Their Implications*** di R.Hill e C.S.Hiremath, oltre ad alcuni capitoli delle dispense del Prof. Monaci (corso di Ricerca Operativa 2).

A questo punto il problema è formalizzato in modo completamente centralizzato e vorremmo valutare se è possibile modellizzarlo in modo tale che possa essere svolto, almeno parzialmente, in modo distribuito. Segue qualche riferimento a materiale riguardante quest'ambito.

- **Luc Brunet.** ***Consensus-Based Auctions for Decentralized Task Assignment.*** **2008**

  Si tratta di una tesi che offre una descrizione generale del problema del task assignment e si focalizza successivamente su di un algoritmo distribuito del tipo Consensus-Based Auction Algorithm.

- **M Alighanbari et al.** ***Decentralized Task Assignment for Unmanned Aerial Vehicles*** **(articolo) e** ***Robust and Decentralized Task Assignment Algorithms for UAVs*** **(Tesi)**

  Successivamente, ci sono parsi interessanti alcuni lavori riguardanti tecniche per il task assignment decentralizzato per UAV (Unmanned Aerial Vehicles), benchè i metodi proposti non siano immediatamente riformulabili in modo tale da essere applicabili al problema delle telecamere. In particolare la difficoltà emerge nel definire un'adeguata funzione di costo e nel pesare il vincolo che nel caso di interesse non tutti i task sono eseguibili da tutti gli agenti (problema delle aree visive). Anche in questo caso il problema è formulato come variante di KP (precisamente del MMKP - multiple choice multi-dimensional KP).

- **A study of Coordinated Dynamic Market-Based Task Assignment in Massively Multi-Agent Systems**

  Si tratta di uno degli articoli esaminati riguardanti gli algoritmi market-based per l'assegnazione di task ad agenti. Potrebbe essere utile qualora riuscissimo ad attuare una politica di assegnazione distribuita. L'articolo analizza vari metodi cooperativi ed uno non cooperativo, illustrando diversi meccanismi d'asta. Sono proposte diverse soluzioni per la costruzione di funzioni di utilità e costo ed un intelligente meccanismo di scambio dei task tra agenti, nel caso sia vantaggioso globalmente.

- **Brent Lagesse.** ***A Game-Theoretical Model for Task Assignment in Project Management*** **. 2006**

  Approccio basato sulla teoria dei giochi che mira ad ottenere match tra task ed agenti, tenendo conto di tempo, preferenze ed abilità di questi ultimi.

  1. Idea di partenza: algoritmo di Gale-Shapely (problema dei matrimoni stabili: si dice che il match è instabile se, date le coppie (m,w) ed (m',w'), m preferisce w' a w e w' preferisce m ad m'), che tiene conto delle preferenze degli elementi di due insiemi, che vanno combinati. Problemi da risolvere: GS non assicura che tutti i task vengano assegnati.
  2. Strategia: si assegnano pesi ai task (che rappresentano il tempo richiesto) ed uno slot temporale da riempire agli agenti; si introducono callbacks per verificare l'effetto del rifiuto dei task.
  3. Ai task viene assegnata una struttura comprensiva di lista di preferenze, tempo richiesto, puntatore alla preferenza non rifiutata successiva; la struttura assegnata agli agenti comprende anch'essa una lista di preferenza ed il puntatore alla successiva preferenza non rifiutata. Ne fanno parte, inoltre, un indice del tempo disponibili ed una lista di callback.
  4. L'algoritmo prevede che un task si rivolga al suo agente preferito, che se lo preferisce al task che sta eseguendo, lo sostituisce ad esso. Qualora l'agente non potesse più,per limite di tempo disponibile, eseguire il task precedente, questo si "libera" lanciando un callback, che altri agenti potrebbero risolvere in seguito. La procedura garantisce l'assenza di deadlock (grazie alle liste di precedenza, secondo gli autori) e la terminazione (task tutti processati -al massimo rifiutati da ogni agente)

- Parallelamente procede la costruzione di un simulatore. I task vengono generati in modo semplice, assumendo che siano processi indipendenti di Poisson (tempi di interarrivo sono v.a. esponenziali). Confluiscono in un pool globale, noto a cisacun agente (ipotesi compatibile con le caratteristiche del sistema da noi definito), dal quale vengono estratti quando scelti da una delle telecamere. Il

meccanismo di scelta implementa una logica *greedy* ed in un primo momento completamente deterministica, che privilegia task con priorità maggiore (come il brandeggio comandato dall'operatore). La mancanza di conoscenza sui tempi di interarrivo medi del sistema reale è un ostacolo all'analisi delle prestazioni. Le nostre assunzioni sono:

1. *Tempi medi di occorrenza (in s)*:
   Mop = 3600; *Tempo medio di occorrenza del task operatore*
   Mtrk = 3600; *Tempo medio di occorrenza del task tracking*
   Mstr = 300; *Tempo medio di occorrenza del task streaming*

2. *Tempi medi di servizio (in s):*
   Sop = 300; *Tempo medio di occorrenza del task operatore*
   Strk = 600; *Tempo medio di occorrenza del task tracking*
   Sstr = Mstr/5; *Tempo medio di occorrenza del task streaming*

3. Tempo di simulazione: una giornata (24×3600 s)
4. 8 Agenti e 9 Aree visive disgiunte
5. Matrice di copertura:
$$V = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$
6. Matrice di adiacenza:
$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

L'algoritmo procede facendo scegliere ciclicamente agli agenti se scambiare l'eventuale task in esecuzione con uno del pool. Sono state introdotti dei vincoli per cercare di ottenere, per quanto possibile, che in ciscuna area visiva almeno una telecamera registri video in tempo reale (operazione incompatibile con il task di *streaming*). Si nota che i task di streaming tendenzialmente vengono accodati, essendo a bassa priorità. Vorremmo cercare di modulare in modo dinamico la priorità per evitare che questa tipologia di task venga disattesa nella maggioranza dei casi. Altrimenti, allo scadere di un certo intervallo successivo alla generazione dei task streaming, essi potrebbero essere parzialmente soddisfatti con la trasmissione, molto meno onerosa, di una o più immagini snapshot.