Metodi e Modelli per l'Ottimizzazione Combinatoria

Progetto: Metodo di soluzione euristico

Luigi De Giovanni

Viene presentato uno schema di soluzione euristica per la configurazione ottimale di reti ad-hoc secondo il modello multi-server. L'euristica si basa sulla soluzione dei modelli già implementati in Cplex come sottoproblema.

ATTENZIONE!

- **La scadenza** ultima è fissata per il 07/01/2013 alle ore 9:00 am.
- Per chi vuole sostenere **l'orale prima delle vacanze di Natale**, la scadenza è fissata a due giorni prima della data concordata con il docente per l'orale: ad esempio, per sostenere l'esame martedì 18, dovreste consegnare entro domenica 16. Mi raccomando, ritagliatevi del tempo per preparare l'orale...
- Per agevolare l'implementazione, potete utilizzare il vostro codice, anche se non fornisce i risultati "corretti". SE AVETE EFFETTUATO NEI TERMINI LA CONSEGNA DELLA FASE 3 ma il vostro codice non è funzionante (ad esempio, ricorrenti segmentation fault che vi impediscono di lavorare), il docente potrà mettere a disposizione, su richiesta, un codice (maggiormente) funzionante.
- Nel prosieguo è indicata una parte opzionale (istanze da voi generate, funzione di valutazione [vedi sotto]): si può prendere 30 e lode anche senza implementarla (soprattutto se si sostiene l'esame nella prima sessione), ma la sua presenza aiuta, soprattutto per chi consegna il progetto complessivo dopo la scadenza del 07/01/2013.

.

La prossima (quarta) consegna è relativa alla consegna via email di un file compresso contente:

- la vostra implementazione in C++ (con l'utilizzo delle API di Cplex) del metodo euristico discusso in questo documento. Si richiedono sorgenti e makefile che siano compilabili su una macchina linux del laboratorio LabTA (quindi con i path del laboratorio).
- un file pdf con un relazione sul lavoro svolto contenente:
  - la descrizione dei metodi implementati (non a livello di codice, ma metodologico);
  - la descrizione delle implementazioni (qualche dettaglio che guidi la lettura del codice);
  - la descrizione dei risultati computazionali, come sotto indicati.

1 Euristica costruttiva

Con riferimento alla notazione adottata in precedenza, ricordiamo che il problema consiste nello scegliere, tra quelli disponibili nell'insieme N, L nodi che funzionino, a turno, da server e, per ogni turno, stabilire la strategia di routing, in modo da massimizzare la durata del gioco.

Per massimizzare la durata del gioco dovremmo, euristicamente, cercare di garantire che, ad ogni turno, il consumo di energia sia minimo, almeno nei nodi critici. Diciamo che un nodo è tanto più "desiderabile" quanto più riesce a garantire, se agisce come server, tale consumo minimo nei nodi critici. In altre parole, un nodo è più desiderabile se, funzionando come unico server, garantisce una maggiore durata della rete.

Un possibile algoritmo euristico è quindi il seguente:

1. Per ogni nodo $v \in N$, calcolare una misura di desiderabilità Φ_v.
2. Scegliere gli L nodi con Φ più elevata come server.
3. Determinare la durata e il routing ottimale.

Per implementare il passo 1, consideriamo il modello presentato in precedenza per il caso con un solo server (o, equivalentemente il modello multiserver con $L = 1$). Una volta fissato il nodo che agisce da server (variabili σ), il modello diventa un modello di Programmazione Lineare (senza variabili intere o binarie) e, quindi, possiamo sperare di risolverlo in modo relativamente efficiente, con tempi ragionevoli. Pertanto, dato un nodo $v \in N$ possiamo calcolare una misura di desiderabilità Φ_v in questo modo:

(i) si fissa $\sigma_v = 1$ (e tutti gli altri a 0);

(ii) si risolve il modello con le σ fissate. Questo punto si può implementare in due modi alternativi:

  - fissando, mentre si crea il modello, a 1 il lower bound di σ_v, e a 0 l'upper bound di $\sigma_i, i \neq v$;
  - oppure evitando di creare le variabili σ e creando i vincoli considerando direttamente la sostituzione dei valori 0 o 1 per i diversi σ (questo metodo è più efficiente, ma richiede una modifica delle procedure di creazione del modello, quindi NON è richiesto, neanche come opzionale).

(iii) si legge il valore della funzione obiettivo e si pone Φ_v pari a questo valore.

Per quanto riguarda il passo 3, questo può essere implementato facilmente se si ha a disposizione il modello multiserver, mentre richiede delle preelaborazioni nel caso si utilizzi il modello con un solo server.

Pertanto, l'implementazione di questo passo è OPZIONALE.

Nel caso di modello multiserver, basta eseguire il modello stesso avendo fissato a 1 (come detto sopra) i valori delle L variabili σ corrispondenti ai nodi scelti al passo 2.

Se non si ha a disposizione il modello multiserver, per ottenere una stima euristica della durata e del routing, si può procedere come segue. Sia N_L l'insieme degli L nodi server selezionati al passo 2.

- Per ciascuno nodo $w \in N_L$
  - creare il modello considerando che
    - si fissa σ_w a 1;
    - nei vincoli (3), se $i \in N_L$, si deve sommare anche il consumo P_i^D;
    - nei vincoli (7), quando $v \in N_L$, si utilizza come coefficiente delle variabili y il valore $(B_v^S \cdot D_v^S + B_v^D \cdot D_v^D)$, per considerare anche la banda del traffico dal server attivo (cioè w) al server dormiente v;
    - nei vincoli (8), quando $v \in N_L$, si utilizza come coefficiente delle variabili y il valore $(J_{ikv}^{SR} \cdot B_v^S + J_{ikv}^{DR} \cdot B_v^D)$, per considerare anche i consumi per il traffico dal server attivo (cioè w) al server dormiente v;
    - nei vincoli (9), quando $v \in N_L$, si utilizza come coefficiente delle variabili y il valore $(J_{ikv}^{ST} \cdot B_v^S + J_{ikv}^{DT} \cdot B_v^D)$, per considerare anche i consumi per il traffico dal server attivo (cioè w) al server dormiente v;
  - risolvere il modello
  - per ogni nodo $i \in N$, leggere il valore della variabile ζ_i e memorizzarlo come $\zeta_i^{(w)}$, cioè il consumo registrato per il nodo i nel turno di w.
  - memorizzare le variabili x e y come $x^{(w)}$ e $y^{(w)}$ (si tratta della strategie di routing quando il server attivo è w)
- Per ogni nodo $i \in N$, calcolare il consumo medio $\bar{\zeta}_i = \sum_{w \in N_L} \zeta_i^{(w)}$

- calcolare il valore $\delta = \min_{i \in N} \frac{\bar{\zeta}_i}{A_i}$ (si tratta della durata del nodo che consuma per primo la batteria, quindi la durata del gioco)
- restituire le strategie di routing memorizzate $x^{(w)}$ e $y^{(w)}$ e la durata δ.

Solo per completezza espositiva di quanto risportato in laboratorio (e quindi senza richiesta, neanche opzionale, di implementazione), si fa notare che, avendo a disposizione una procedura (una di quelle sopra esposte) per la valutazione della soluzione quando i server sono fissati, è facile pensare a una possibile metaeuristica, ad esempio di tipo neighbourhood search, in cui:

- la soluzione iniziale si ottiene con l'euristica descritta;
- la rappresentazione della soluzione è un vettore di L componenti riportante i nodi considerati server (o una stringa binaria di $|N|$ elementi che valgono 1 se il corrispondente nodo è server, 0 altrimenti);
- le mosse sono di scambio di un nodo server con uno non-server;
- la valutazione delle soluzioni vicine si ottiene usando la procedura di cui sopra.

2 Prove computazionali

Le prove computazionali consistono nel far girare le procedure di ottimizzazione su delle istanze. In particolare sono fornite delle istanze tramite il link "Istanze" (potete metterle, se preferite, nel vostro formato di input). Si consiglia di considerare un tempo limite di 3600 secondi (le istanze più grandi richiedono molto –molto– più tempo per essere risolte all'ottimo).

> Potete anche generare delle altre istanze, in cui varino, oltre alla posizione dei nodi, anche alcuni parametri, come la carica iniziale, i consumi per le trasmissioni etc. **Questa parte è opzionale**.

I risultati devono essere espressi in termini di valore della soluzione e tempi di calcolo per il modello della fase 3, e riportare sinteticamente, se diponibile (vedi parte opzionale nella sezione precedente), un confronto con gli stessi risultati per il metodo euristico.