Corso di Metodi e Modelli per l'Ottimizzazione Combinatoria - Relazione progetto

Universitá degli studi di Padova
Anno Accademico 2015/2016

Studente: Giacomo Quadrio
Matricola: 1061566

Sommario

Il progetto del corso di Metodi e Modelli per l'Ottimizzazione Combinatoria consiste nel risolvere un problema che vede coinvolta un'azienda metalmeccanica che produce pannelli forati per la costruzione di quadri elettrici. La foratura di questi è eseguita attraverso un macchinario a controllo numerico dotato di una punta diamantata che, muovendosi sul pannello secondo una sequenza programmata, produce i fori nelle posizioni desiderate. L'obiettivo è quindi quello di individuare la sequenza di foratura ottimale che minimizzi i tempi di produzione, tenendo conto che il tempo necessario per la foratura è lo stesso e costante per tutti i punti.

1. Modello del problema

Il problema oggetto del progetto puó essere formulato come un problema di ottimizzazione su reti di flusso partendo quindi da un grafo G = (N,A). Scegliendo arbitrariamente un nodo di partenza $0 \in N$ impostiamo ad |N| il flusso uscente da esso in modo tale che venga spinto verso altri nodi. Tale operazione ha peró dei vincoli ovvero ciascun nodo, eccetto l'origine, riceverá una e una sola unitá di flusso, ogni nodo sia visitato una e una sola volta e che il costo del cammino, in termini di pesi c_{ij}, sia minimo.

1.1. Il modello nella Programmazione Lineare Intera

Il problema può essere formalizzato con il seguente modello di programmazione lineare intera. Avremo quindi:

Insiemi

- **N:** nodi del grafo, rappresentano le posizioni dei fori da realizzare
- **A:** insieme degli archi (i,j) con i e j $\in$ N. Essi rappresentano il tragitto per spostarsi dal nodo i al nodo j

Parametri

- $\mathbf{c_{ij}}$**:** tempo impiegato per spostarsi dal nodo i al nodo j con i e j $\in$ N e l'arco (i,j) $\in$ A.
- **0:** nodo di partenza del cammino $\in$ N.

Variabili decisionali

- $\mathbf{x_{ij}}$**:** unità di flusso trasportate da i a j con i e j $\in$ N e l'arco (i,j) $\in$ A.
- $\mathbf{y_{ij}}$**:** indica l'utilizzo dell'arco (i,j), 1 se viene utilizzato, 0 altrimenti. Avremo che i e j $\in$ N e l'arco (i,j) $\in$ A.

Vincoli

Il modello, a questo punto, prevede un totale di cinque vincoli differenti che possono essere indicati come segue:

1. Il flusso uscente da x_{0j} deve essere massimo, cioè |N|
2. Ogni nodo utilizza al massimo una unità di flusso, tranne il nodo di partenza
3. Ogni nodo ha un solo arco in entrata
4. Ogni nodo ha un solo arco in uscita
5. Se vi è un'unità di flusso trasportata da i a j deve di conseguenza esserci un arco che va da i a j

Modello

Inserire formule varie

2. Metaeuristiche scelte per il modello e loro implementazione

Il progetto da svolgere richiesto dal corso di Metodi e Modelli per l'Ottimizzazione Combinatoria prevede l'implementazione del modello di programmazione lineare intera tramite due tecniche ovvero CPLEX ed una o più metaeuristiche a nostra scelta. Una volta fatto ciò si procederà testando i metodi con delle istanze di prova ed i risultati e statistiche confrontati tra di loro per valutarne le prestazioni.

Nello specifico, il problema in esame è un problema di ricerca di vicinato e consiste nel definire una soluzione iniziale e cercare di migliorarla esplorando un intorno di questa soluzione; quindi i metodi utilizzati all'interno del progetto sono due, la Local Search ed il Simulated Annealing, i cui algoritmi sono riportati qui di seguito:

Codice 1: Local Search

```
sol = getInitialSol(random);

while (true){
        vector<int> newSol = findBestN(sol);
        if (evaluate(newSol) >= evaluate(sol)){
                return evaluate(sol);
        }else{
                sol = newSol;
        }
}
```

Come si può vedere, l'algoritmo di Local Search crea innanzitutto una soluzione iniziale da cui partire tramite la funzione getInitialSol, dopodiché esso è stato implementato utilizzando un unico ciclo while che opera finché non viene restituito in output un valore. Questo significa che ciò avverrà unicamente quando, dopo aver calcolato un una nuova soluzione con newSol = findBestN(sol), il suo valore sarà peggiore del valore della soluzione corrente. Se così non è la soluzione corrente viene aggiornata alla soluzione appena calcolata ed il ciclo continua ad operare fino al punto in cui viene trovata una soluzione peggiorativa.

Codice 2: Simulated Annealing

```
sol = getInitialSol(random);
```

```
n_passi = 100000.0 * n / 5;

while (step < n_passi){
        newSol = getNeigh(sol,2,true);
        float de = evaluate(sol) - evaluate(newSol);
        if (de > 0){
                sol = newSol;
        }else{
                temp = 1-(step/n_passi);
                double prob = exp((de)/temp);
                srand(time(NULL)+for_random);
                for_random = for_random + 1;

                if (prob*100 > (rand()%100) ){
                        sol = newSol;
                }
        }
step ++;
}

return(LocalSearch(sol));
```

Per quanto concerne invece l'algoritmo di Simulated Annealing, anch'esso crea innanzitutto una soluzione iniziale da cui partire tramite la funzione getInitialSol, dopodiché troviamo un ciclo while principale che opererà finché step <= n-passi; da notare che il numero di passi inoltre è dinamico così che cresca al crescere del numero di nodi coinvolti nel problema. In cosa consiste quindi questo algoritmo? In sostanza viene calcolata una nuova soluzione attraverso la funzione getNeigh, dopodiché verrà calcolata la differenza tra il valore della soluzione corrente e quello della nuova soluzione. Se il delta ottenuto è maggiore di zero significa che la nuova soluzione è migliorativa e quindi aggiornerò di conseguenza sol, se invece così non è procederemo in maniera differente rispetto alla Local Search. Andremo infatti a calcolare per prima cosa la temperatura di raffreddamento, valore che è coinvolto nel calcolo della probabilità di accettare una mossa peggiorativa. Questa probabilità è calcolata come segue:

$$\text{prob} = \exp(\ \delta/\text{t})$$

dove δ è l'entità del peggioramento de e t è la temperatura temp di raffreddamento. Nel caso la probabilità prob sia maggiore di un numero random calcolato attraverso la funzione srand ciò comporterà appunto l'accettare la mossa peggiorativa, altrimenti essa verrà scartata. Come si modifica però la probabilità p? Essa diminuisce al crescere del peggioramento indotto dalla mossa stessa e cresce al crescere della temperatura t di processo.

Al termine delle operazioni eseguite tramite il ciclo while andremo ad effettuare infine una fase di intensificazione tramite l'operazione di Local Search eseguita sulla soluzione migliore calcolata in precedenza. Il motivo di ciò è che la Local Search è una tecnica relativamente economica in quanto a tempi di esecuzione e permette di raffinare ulteriormente la soluzione trovata.

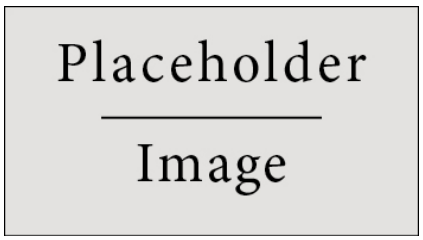

Figura 1: Figure caption

$$e = mc^2 \tag{1}$$

3. Analisi dei risultati ottenuti

Reference to Section 1. Etiam congue sollicitudin diam non porttitor. Etiam turpis nulla, auctor a pretium non, luctus quis ipsum. Fusce pretium gravida libero non accumsan. Donec eget augue ut nulla placerat hendrerit ac ut mi. Phasellus euismod ornare mollis. Proin tempus fringilla ultricies. Donec pretium feugiat libero quis convallis. Nam interdum ante sed magna congue eu semper tellus sagittis. Curabitur eu augue elit.

Aenean eleifend purus et massa consequat facilisis. Etiam volutpat placerat dignissim. Ut nec nibh nulla. Aliquam erat volutpat. Nam at massa velit, eu malesuada augue. Maecenas sit amet nunc mauris. Maecenas eu ligula quis turpis molestie elementum nec at est. Sed adipiscing neque ac sapien viverra sit amet vestibulum arcu rhoncus.

Vivamus pharetra nibh in orci euismod congue. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Quisque lacus diam, congue vel laoreet id, iaculis eu sapien. In id risus ac leo pellentesque pellentesque et in dui. Etiam tincidunt quam ut ante vestibulum ultricies. Nam at rutrum lectus. Aenean non justo tortor, nec mattis justo. Aliquam erat volutpat. Nullam ac viverra augue. In tempus venenatis nibh quis semper. Maecenas ac nisl eu ligula dictum lobortis. Sed lacus ante, tempor eu dictum eu, accumsan in velit. Integer accumsan convallis porttitor. Maecenas pretium tincidunt metus sit amet gravida. Maecenas pretium blandit felis, ac interdum ante semper sed.

In auctor ultrices elit, vel feugiat ligula aliquam sed. Curabitur aliquam elit sed dui rhoncus consectetur. Cras elit ipsum, lobortis a tempor at, viverra vitae mi. Cras sed urna sed eros bibendum faucibus. Morbi vel leo orci, vel faucibus orci. Vivamus urna nisl, sodales vitae posuere in, tempus vel tellus. Donec magna est, luctus non commodo sit amet, placerat et enim.