Programmazione Lineare & Modellazione

Immagine che contiene tavolo

Descrizione generata automaticamente

Immagine che contiene testo

Descrizione generata automaticamenteImmagine che contiene testo

Descrizione generata automaticamenteModelli inseriti a livello indicativo (non servono per risolvere tutti i modelli, ma a seconda della natura del problema, servono per ricordarsi “che ci vogliono dei vincoli aggiuntivi non esplicitamente presenti nel problema”, da mia esperienza):

Immagine che contiene testo

Descrizione generata automaticamente

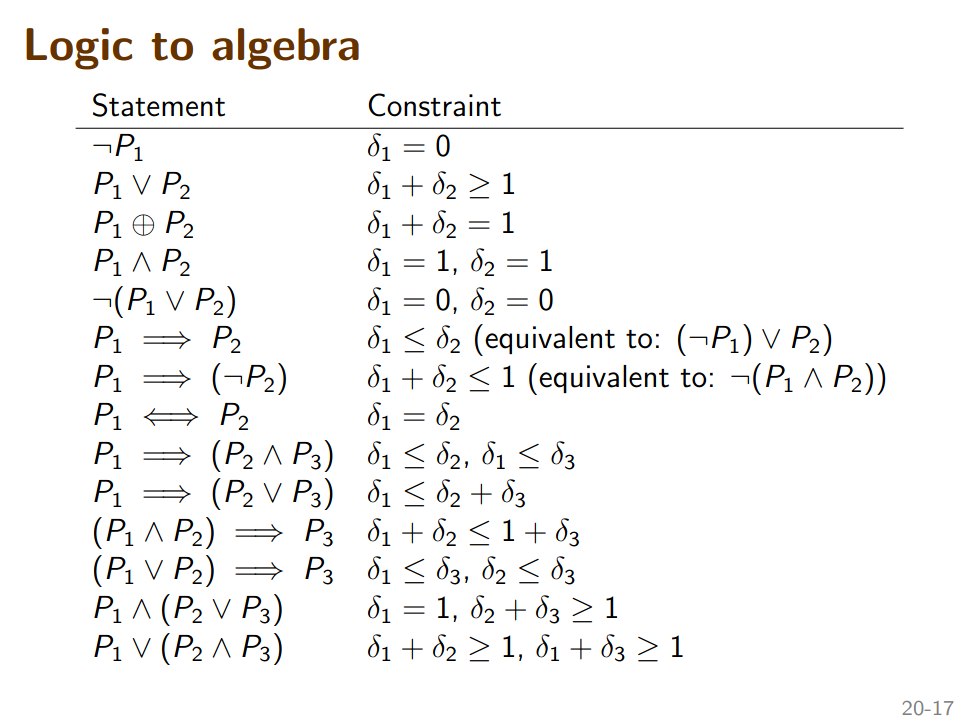
Modello multiperiodale

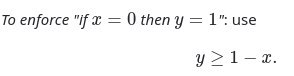
L’effetto di variabili in vincoli precedenti “ritorna” in quelli successivi.

[cfr. Piani di investimento nelle dispense]

Immagine che contiene testo

Descrizione generata automaticamente





Il prof ci tiene particolarmente alla modellazione; fare bene questi esercizi, concentrandosi su questi passi in questo ordine.

Per i 6 crediti:

* No vincoli spuri
* No dimensionamento variabile big M

Passi (occorre ragionare in termini di decisioni da prendere):

1. Individuare quali sono le decisioni da prendere per risolvere il problema, ossia dobbiamo definire le variabili decisionali
   1. Queste variabili possono essere ad uno o a due indici; per esperienza, spesso, le variabili a due indici fanno danni; meglio mettere più variabili ad un indice. Preferibile mettere a due indici quando è evidente dal problema che servono entrambe le colonne di una tabella
   2. Attenzione anche alla scrittura dei vincoli; “indipendentemente da” fa capire che, probabilmente, avremo bisogno di una variabile ad indice solo e non due o dipende dal contesto
2. Scrivere la funzione obiettivo
3. Scrivere almeno uno degli altri vincoli, preferibilmente non gli ultimi. Almeno uno/due tra i primi aggiuntivi si scrivono subito
   1. A detta stessa del prof, almeno uno o due di questi servono per aiutare nella scrittura del modello
4. Scrivere il resto dei vincoli, considerando che:
   1. Se si ha a che fare con vincoli logici del tipo “posso decidere *se fare* la cosa X” va inserita una variabile binaria
      1. Questa variabile va *legata* ad una variabile normale per fare in modo stia in piedi.
      2. Solitamente questo avviene con la forma big-M 🡪
         1. Essi servono a linearizzare un modello evitando i termini quadratici (moltiplicazione o somma di più termini singoli)
   2. Si possono avere diversi casi notevoli, quali:
      1. Costi fissi:
         1. In questo caso, va aggiunto un termine moltiplicativo a tutte le variabili decisionali legate per quantità e si crea una variabile decisionale apposita per costo fisso
      2. Almeno:
         1. Se vincolo logico, usare “insieme di variabili logiche >= numero” e legare le variabili con vincolo di big-M nel modo
         2. Se vincolo normale, usare “variabile decisionale >= numero”
      3. Al massimo:
         1. Se vincolo logico, usare “insieme di variabili logiche <= numero” e legare le variabili con vincolo di big-M nel modo
         2. Se vincolo normale, usare “variabile decisionale <= numero”
      4. Esattamente:
         1. Se vincolo logico, usare “insieme di variabili logiche = numero” e legare le variabili con vincolo di big-M nel modo
      5. Penalità/Sconto
      6. *Scelgo* di pagarla, pertanto sarà una variabile binaria legata a variabili logiche già presenti e poi aggiunte in funzione obiettivo
      7. Budget
         1. Variabili decisionali <= budget
      8. Capacità:
         1. Variabili decisionali <= cap. minima
         2. Variabili decisionali >= cap. massima
5. Scrivere i domini considerando che
   1. Al 90% le variabili sono intere
   2. Si mettono delle variabili reali qualora ci fossero frazioni/radici

Consigli ulteriori:

* se si sta scrivendo un vincolo ma non si riesce a capire come scriverlo, probabilmente può essere scritto in maniera più semplice, anche ripensando le stesse variabili. A me aiuta molto cercare di semplificare e considerare i punti detti.

Simplesso: Esercizi pratici

Si porta in forma standard, cioè

* Funzione obiettivo
  + Se è *max* diventa *min* e cambio tutti i segni; se è già *min* non si cambia nulla
* Vincoli di uguaglianza 🡪 Aggiungo le variabili di slack e la disuguaglianza diventa uguaglianza
  + Se si ha la variabile di slack viene aggiunta come positiva
  + Se si ha la variabile di slack viene aggiunta come negativa
* Variabili non negative (vincolo ) 🡪 Aggiungo variabile positiva e cambio il segno a tutte le occorrenze della stessa variabile (il dominio è nella forma , )
* Termini noti non negativi 🡪 Se un termine noto è negativo, cambio segno a tutta la disuguaglianza

Si eseguono i calcoli, ricordandosi che si devono scrivere alla fine dei passaggi di pivoting per l’iterazione corrente di simplesso, le variabili in base (cioè, nella riga corrispondente alla posizione che si sta attualmente calcolando). Si esegue il pivot rispetto alla colonna individuata nel tableau precedente (operazioni di somma/prodotto, etc.)

Imposto i tableau, ricordando che ogni volta mi chiedo:

* È in forma canonica rispetto alla base corrente? Basta avere le colonne (anche sparse/non in ordine) della matrice identità nel tableau rispetto agli elementi della base
  + Es. se la base è allora avrò gli 1 in corrispondenza di questi elementi in colonna
* È ammissibile? 🡪 Tutti i devono essere positivi (se parto da una base ammissibile, rimango sempre in base ammissibile; mai dire che è impossibile)
* È ottima? 🡪 Devo avere tutti i costi ridotti (in altri termini, tutte le variabili fuori base hanno costi ridotti >= 0)
* È illimitata? 🡪 Mi trovo ad avere un coefficiente nella riga dei costi ridotti con tutti i valori negativi sotto nella stessa colonna (in altri termini, tutte le variabili in base hanno costi ridotti <= 0)
* Entra in base? 🡪 La variabile costo ridotto negativo di indice minimo (Bland); si va per ordine posizionale di indice
  + Es. se devo scegliere tra e scelgo per ordine posizionale di indice
* Esce dalla base 🡪 La variabile che ha rapporto minimo prendendo in colonna la variabile entrante e facendo i rapporti minimi. Come prima, si prende la variabile di indice minimo posizionale (Bland)
  + Es. se entra in base prendo considero tutti i rapporti tra la variabile della colonna di   e gli elementi di questa, quindi su tutta la colonna di e gli elementi in riga corrispondenti
* Ricordarsi che si sostituisce una variabile entrante/uscente esattamente nella posizione da cui è stata tolta
  + Es. se da tolgo per far entrare es. , l’ordine rimane )

Ad ogni passaggio, eseguo il pivoting ogni volta prendendo in colonna la variabile che esce e in riga la variabile che entra.

Quando trovo la soluzione, considero che è tutta la colonna in cui prendo il valore della funzione obiettivo (solitamente e cambio segno per questo motivo); normalmente, il suo valore migliora (o non peggiora) tra i passaggi) e come vincoli dobbiamo fare delle precisazioni, riferite al problema di partenza (noi lo convertiamo in forma standard) ed esclusivamente alle variabili di slack:

* I vincoli laschi, nel momento in cui abbiamo una variabile di slack
  + Si dice anche “soddisfatti col minore/maggiore stretto nella soluzione ottima”
* I vincoli saturi, nel momento in cui abbiamo una variabile di slack pari a 0 (uguale a 0)
  + Si dice anche “soddisfatti all’uguaglianza nella soluzione ottima”

Nella scrittura della soluzione, si deve considerare che nel tableau abbiamo , dunque va invertito il segno. Poi, si scrivono tutte le variabili (es. ho 7 variabili e scriverò con rispettivo valore (positivo se in base, quindi guardo quello che sta in , 0 altrimenti se fuori base sarà pari a 0, distinguendo opportunamente tra saturi e laschi).

Se si parte da un problema di massimo 🡪 la soluzione è . Esempio: soluzione del problema è . La soluzione considera , quindi sarà . Questa però è appunto

Domande teoriche sul simplesso

* Individuazione basi ottime senza operazioni di pivot
  + Basta vedere dove sono gli 1 della matrice identità e dove ci sono costi ridotti non negativi [quindi, sia uguali a 0 che maggiori] e scegliere come base quella che ha i coefficienti più a sinistra
* Scrivere la soluzione di base corrente e dire se è ottima.
  + Basta individuare il valore di (ricordandosi che è ) e scegliere come base la prima per i coefficienti sparsi della matrice identità (cioè, se ho più coefficienti tra i quali poter scegliere per capire la base, scelgo quello più a sinistra)
* Stabilire, SENZA EFFETTUARE LE OPERAZIONI DI PIVOT, quale sarà il valore della funzione obiettivo alla fine della prossima iterazione del simplesso. GIUSTIFICARE LA RISPOSTA!
  + Usiamo la regola di Bland per selezionare la variabile entrante ed eseguiamo il solito rapporto minimo tra e il punto della variabile entrante
    - Se si ha un rapporto minimo pari a 0, la f.o. non migliora
    - Se si ha un rapporto minimo in cui il minimo elemento è positivo, basta eseguire il prodotto tra il coefficiente individuato dall’operazione di rapporto minimo e il costo ridotto della stessa colonna.

* + - Successivamente, si esegue una somma algebrica con il valore della funzione obiettivo (solitamente, si ha
* Alla fine della prossima iterazione sarà cambiata la base corrente: sarà cambiato anche il vertice del poliedro associato alla nuova base? GIUSTIFICARE LA RISPOSTA!
  + Per calcoli, si ragiona che si effettua una sottrazione tra e , per tutte le variabili in base. In base alla regola di Bland e rispetto a chi esce dalla base e letteralmente rifacciamo lo stesso calcolo rispetto alla variabile che entra in base, tenendo le altre. Capiamo così se l vertice è cambiato o meno.
  + Letteralmente, se ho due variabili con stesso rapporto minimo, allora so che l’iterazione sarà degenere e il vertice del poliedro non cambia valore
  + Se invece ho che non ho due variabili con rapporto minimo, allora l’iterazione non sarà degenere e quindi il vertice del poliedro cambia valore; questo succede perché si ha una variabile a costo ridotto negativo che consente di migliorare la base corrente.
* Riusciamo ad individuare una soluzione di base corrispondente? Quale?
  + Il tableau è in forma canonica e individuo come soluzione di base una tra quelle in cui si hanno i coefficienti della matrice identità. Inoltre, occorre dire quanto vale e quanto vale (cioè, quanto valgono tutti i coefficienti dentro e fuori dalla base ammissibile)
  + Perché non è ottima?
    - Non sappiamo se sia ottima, avremo dei coefficienti di costo ridotto negativo (condizione sufficiente ma non necessaria per ottimalità)
  + È ottima?
    - Basterà avere tutti coefficienti di costo ridotto positivo; in altri termini, tutte le variabili fuori base hanno costi ridotti positivi
* Perché la teoria del simplesso non consente l’operazione di pivot sull’elemento X?
  + L’operazione non è consentita, in quanto quell’elemento non corrisponde al rapporto minimo
  + In altre parole, la variabile corrispondente alla colonna su cui si trova l’elemento riquadrato su cui non si può fare pivot, assumerebbe un valore tale da portare a 0 la corrispondente variabile alla riga su cui si trova l’elemento riquadrato su cui non si può fare pivot con un valore troppo alto per soddisfare i restanti vincoli e le altre variabili dovrebbero assumere valori negativi
* Su quali elementi è possibile effettuare il pivot secondo le regole del simplesso (indipendentemente da regole anticiclo)?
  + Prendendo come elementi la variabile che entra e la variabile che esce a prescindere da Bland, quindi letteralmente tutte le variabili che per riga/colonna corrispondono al rapporto minimo
* Considerando le variabili ordinate per indice crescente, quale sarà il cambio base secondo le regole del simplesso e applicando Bland?
  + Si considerano le regole dette (variabile che entra/variabile che esce secondo quanto scritto sopra) e poi prendo le variabili con indice posizionale minore (secondo Bland)
* (Non una domanda teorica, ma presente negli esercizi di calcoli del simplesso); in base a quale teorema è possibile determinare direttamente il valore della soluzione ottima?
  + Se il problema primale è illimitato
    - il problema duale è inammissibile (corollario del teorema della dualità debole)
  + Se il problema primale ammette soluzione ottima
    - Il corrispondente problema duale ha soluzione ottima finita e i valori coincidono (teorema della dualità forte)
* Supponiamo di effettuare un cambio base in cui entra in base la variabile X: perché la soluzione di base ottenuta a seguito di questo cambio base è sicuramente degenere?
  + In questo caso, avremmo X righe corrispondenti al rapporto minimo, pertanto avremo che almeno una esce dalla base con valore 0, mentre le altre Y assumeranno valore 0 rimanendo in base.

In limitati casi potrebbe essere che venga indicato di risolvere il problema di PL partendo da una base.

Quello che si fa è:

* Portare il problema in forma standard
* Considerare una delle due basi organizzando i dati in forma tableau tralasciando la f.o.
* Eseguire le operazione di pivot rispetto al primo elemento della base
  + Ad esempio, se devo scrivere la cosa rispetto alla base considero come elemento di pivot
  + Ad esempio, se devo scrivere la cosa rispetto alla base considero come elemento di pivot
* Se riesco a seguito di tutto il pivot ad ottenere una soluzione ammissibile, posso applicare il metodo del simplesso e risolvere.

Allego esempio grafico di quanto appena detto per chiarezza.

Immagine che contiene testo, tavolo

Descrizione generata automaticamenteImmagine che contiene tavolo

Descrizione generata automaticamente

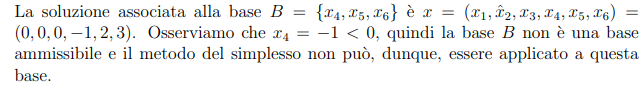


Immagine che contiene testo

Descrizione generata automaticamente Condizioni primale/duale e passaggio al problema duale

Immagine che contiene tavolo

Descrizione generata automaticamente(se problema di massimo va letta a rovescio)

Enunciare le condizioni di complementarietà primale-duale

Immagine che contiene testo

Descrizione generata automaticamente

Dualità

Abbiamo il problema e la verifica di una certa soluzione. Quindi:

1. Verifica ammissibilità primale dela soluzione data 🡪 Sostituisco i valori della soluzione che mi viene data dentro i vincoli e verifico se sono rispettate tutte le disuguaglianze
2. Passaggio al problema duale
   1. Se ho problema di min passo a problema di max (o viceversa)
   2. Considero come termini tutti i termini noti (compresi i loro coefficienti)
   3. Per ogni vincolo, prendo tutti le variabili con coefficienti alla colonna corrispondente alla posizione , e:
      1. Quando passo da a riporto l’opposto del segno della corrispondente variabile di dominio primale
      2. Quando passo da a , riporto lo stesso segno della corrispondente variabile di dominio primale
      3. se non c’è nulla per la variabile quando si scrive il vincolo duale, si vede come (questo anche per quando si deve trovare l’opposto tra min-max o max-min; l’opposto di una uguaglianza è sempre una uguaglianza)
   4. Si inseriscono i domini delle variabili, considerando che:
      1. Se passo da problema di a problema di , il dominio delle duali corrisponde allo stesso segno di uguaglianze/disuguaglianze delle variabili primali
      2. Se passo da problema di a problema di , il dominio delle duali corrisponde all’opposto del segno di uguaglianze/disuguaglianze delle variabili primali
3. CCPD - Applicazione delle condizioni di complementarietà primale-duale
   1. Primo pezzo: vincoli primali
      1. Prendo posizione della riga attuale e lo moltiplico per il vincolo primale in posizione portando a sinistra il termine noto (cambiando quindi di segno)

Es.

* + 1. Sostituisco quindi i valori della soluzione iniziale data nel vincolo e:
       1. Se ho un valore , allora questo viene considerato come condizione
       2. Se ho un valore , allora non mi dice nulla e non lo considero (*non posso dedurre condizioni*)
    2. Se ho vincoli di uguaglianza non posso dire nulla (*deriva dall’ammissibilità primale* *e non posso dedurre condizioni di complementarietà per la variabile ) 🡪 no CCPD*
  1. Secondo pezzo: vincoli duali
     1. Prendo posizione della riga attuale e lo moltiplico per il vincolo duale in posizione portando a sinistra il termine noto (cambiando quindi di segno)

Es.

* + 1. Sostituisco il valore di in quella posizione e faccio le stesse verifiche dei sottocasi (1) e (2) della seconda condizione del problema primale
    2. Se ho vincoli di uguaglianza, devo verificare che non faccia già parte dei vincoli; nel qual caso lo considero (*deriva dall’ammissibilità duale*), altrimenti no

1. Sistema di equazioni CCPD e ammissibilità duale - Sistema di equazioni per l’imposizione delle condizioni di complementarietà primale duale (ccpd) trovate e delle condizioni di ammissibilità duale:
   1. Metto insieme tutte le condizioni che ho trovato fino ad ora e vado a sostituire i valori di oppure di che ho già dal sistema di equazioni e risolvo trovando tutte le soluzioni
2. Verifica ammissibilità duale
   1. Verifico se tutti i valori di soddisfano i vincoli duali (cioè, soddisfano le disuguaglianze)
   2. Verifico se tutti i valori di soddisfano il dominio duale (es. soddisfa )
3. Conclusioni
   1. (Happy Ending)
      1. è ammissibile primale (come da verifica)
      2. è ammissibile duale (come da costruzione e da verifica)
      3. sono in scarti complementari
      4. Le due soluzioni sono ottime per i rispettivi problemi primale e duale
      5. Per verifica si confrontino i valori delle f.o. dei problemi primale e duale; saranno uguali per il teorema della dualità forte
   2. (Bad Ending)
      1. La soluzione trovata è l’unica soluzione del sistema di cui al punto 4 e, quindi, l’unica che soddisfa i vincoli duali di uguaglianza e che è in scarti complementari con la soluzione primale data. Tale soluzione però non è ammissibile per il problema duale. Pertanto non è possibile trovare nessuna soluzione ammissibile duale che sia in scarti complementari con la soluzione primale data, che, quindi, non è ottima

Branch and Bound 🡪 NO Esercizi dello Zaino (0-1)

Minimo 🡪 [LB; S.A.]

Massimo 🡪 [S.A.; UB]

1) Individuare se si tratta di problema di minimo o di massimo

* Se si tratta di problema di minimo i LB aumentano (o non decrescono) di padre in figlio
  + In questo caso avremo come struttura [LB; S.A]
* Se si tratta di problema di massimo gli UB decrescono (o non crescono) di padre in figlio
  + In questo caso avremo come struttura [S.A.; UB]

2) Individuare nodi da poter chiudere

In generale, non chiudo i nodi che sono già sviluppati (quindi normalmente, il nodo radice e nodi che hanno dei figli) e poi:

* Se si tratta di problema di minimo
  + Chiudo i nodi che hanno un LB >= S.A (tengo i nodi con LB < S.A)
* Se si tratta di problema di massimo
  + Chiudo i nodi che hanno un UB <= S.A (tengo i nodi con UB > S.A.)

3) Intervallo ottimo / Intervallo in cui è sicuramente compreso il valore della f.o. / Miglior valore per una soluzione ammissibile

* Se si tratta di problema di minimo
  + Considero il miglior UB (minimo) tra tutti i nodi (attuale soluzione ammissibile/incumbent) e come LB il minore tra i nodi aperti (quindi, non )
* Se si tratta di problema di massimo
  + Considero il miglior LB (massimo) tra tutti i nodi (attuale soluzione ammissibile/incumbent) e come UB il maggiore tra i nodi aperti (quindi, non )
* Se si parla di miglior valore come soluzione ammissibile 🡪 Si cerca minimo-massimo tra tutti i possibili nodi
* Se si parla di miglior valore come valore ottimo 🡪 Si cerca minimo-massimo tra i soli nodi aperti

4) Quale sarà il nodo sviluppato per primo in una strategia Best Bound First?

* Se si tratta di problema di minimo
  + Si sceglie il nodo con il miglior LB (quello minimo) tra i nodi aperti
* Se si tratta di problema di massimo
  + Si sceglie il nodo con il miglior UB (quello massimo) tra i nodi aperti

5) Si supponga che lo sviluppo di cui al punto precedente porti a due nodi figli, di cui uno è relativo ad un insieme di soluzioni vuoto. Si dia un esempio di valori di LB e UB relativi al secondo nodo, che consentano di riconoscere subito la soluzione ottima del problema.

* Chiamiamo il nodo aperto per esempio ; la selezione viene fatta solo nei nodi tuttora aperti, compreso . Si deve considerare come figlio del nodo best bound first e che porta ad una soluzione non ammissibile (può essere anche e , la sostanza è avere due nodi).
  + Se per un problema di minimo, dobbiamo prendere un LB che rispetti la proprietà padre-figlio (quindi >= LB del nodo padre), mentre prendo come UB una nuova incumbent, cioè un valore che sia <= a tutti i LB presenti
  + Se per un problema di massimo, Il LB deve essere una nuova incumbent, dunque >= dei nodi che si vogliono chiudere (quindi, maggiore al loro UB), mentre l’UB deve essere compatibile con il fatto di essere figlio del nodo best-bound first, quindi essere <= UB del nodo padre
* Il valore deve essere tale da permettere la chiusura anche del nodo , quindi possibilmente dentro l’intervallo UB/LB individuato
* Normalmente, si può avere lo stesso valore per UB e LB per semplicità (cosa comune)

6) Individuare possibili valori per UB per mantenere la coerenza con problema di massimo (vuol dire che si avrà “UB?” sul testo)

* Se si tratta di problema di massimo, gli UB decrescono (o non crescono) di padre in figlio; quindi, l’UB dovrà essere compreso tra l’UB del nodo padre (come estremo superiore) e l’UB massimo tra i nodi figli

7) Individuare possibili valori per LB per mantenere la coerenza con problema di minimo (vuol dire che si avrà “LB?” sul testo)

* Se si tratta di problema di minimo, i LB crescono (o non decrescono) di padre in figlio; quindi, il LB dovrà essere compreso tra il LB del nodo padre (come estremo inferiore) e il LB minimo tra i nodi figli

Grafi

Domande:

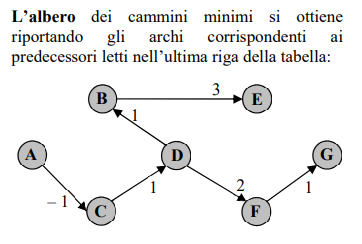
* Si scelga l’algoritmo da utilizzare e si motivi la scelta:
  + Quando si ha un massimo numero di archi/hop (anche se i costi sono tutti positivi) 🡪 Bellman-Ford
    - Posso applicare solo l’algoritmo di Bellman-Ford che è l’unico che dia la possibilità di calcolare i cammini minimi con il massimo numero di archi. Infatti, è possibile dimostrare che, all’iterazione k dell’algoritmo, le etichette corrispondono ai cammini minimi che utilizzano al più k archi. Applicheremo quindi Bellman-Ford fermandoci alla k-esima iterazione, dopo l’inizializzazione.
  + Quando si hanno costi ridotti negativi 🡪 Bellman-Ford
    - Si sceglie l’algoritmo di Bellman­Ford in quanto esistono archi con costo negativo. Bellman-Ford è l’unico algoritmo visto in grado di garantire convergenza alla soluzione ottima del problema dei cammini minimi in presenza di archi di costo negativo, sebbene mediamente meno efficiente dell’algoritmo di Dijkstra, che non garantisce di trovare la soluzione al problema dei cammini minimi se

esistono archi di costo negativo.

* + Se costi ridotti tutti positivi e non ci sono max hop 🡪 Dijkstra, perché più efficiente computazionalmente
    - Essendo tutti i costi sugli archi non negativi, possiamo applicare l’algoritmo di Dijkstra, il più efficiente tra quelli visti

Albero dei cammini minimi:

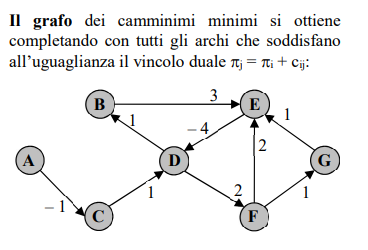
Archi corrispondenti ai predecessori dell’ultima riga; in altre parole, guardo l’iterazione rispetto all’ultima volta in cui un particolare nodo è stato aggiornato, prendendone i predecessori.

Immagine che contiene tavolo

Descrizione generata automaticamente

Grafo dei cammini minimi:

Tutti i cammini minimi; questo vuol dire almeno tutti gli archi dell’albero più tutti quelli ammissibili a costo <= rispetto all’albero (etichette ottime)



* Si riporti un cammino minimo con al più X archi da Y a Z

Si costruisce la catena dei predecessori e si verifica la costruzione del cammino; normalmente, questo viene fatto

* È possibile, basandosi solo sulla tabella precedente e senza fare calcoli, determinare l’esistenza di un ciclo negativo?

Basandosi solo sulla tabella e senza ulteriori calcoli, è possibile determinare il ciclo negativo perché siamo arrivati fino in fondo ai calcoli con etichette instabili (quindi, con nodi appena aggiornati o su cui si ha avuto almeno un aggiornamento).

Fatti:

* Quante iterazioni fare? Tante quante il numero di nodi.
* Non possiamo parlare di albero dei cammini minimi qualora si parla di massimo numero di archi; possiamo invece individuare dei cammini minimi (nel senso che potremmo avere cammini a costo inferiore che però hanno più archi e questo non vale).
* Se la lista dei nodi aggiornati è vuota, le etichette sono stabili e pertanto è possibile determinare albero o grafo dei cammini minimi; altrimenti no.
* Riportare i passi dell’algoritmo scelto in una tabella e giustificarne i passi.

Bellman-Ford

*Indicazioni teoriche (da scrivere in esame)*

Si utilizza una tabella che riporta una riga per ogni iterazione dell’algoritmo. Ogni colonna della tabella è dedicata ad un nodo e riporta, iterazione dopo iterazione, l’evoluzione delle rispettive etichette. L’ultima colonna riporta i nodi aggiornati nel corso dell’iterazione: all’iterazione successiva è sufficiente controllare solo gli archi uscenti da questi nodi (vincolo di Bellman).

La tabella riporta al riga 0 di inizializzazione e una riga per ogni iterazione. All’iterazione si controllano gli archi uscenti da ciascun nodo i nella colonna alla riga , e si aggiornano i costi e i predecessori del nodo all’iterazione qualora l’etichetta del nodo all’iterazione più il costo dell’arco sia strettamente minore dell’etichetta corrente del nodo .

L’algoritmo si ferma qualora la lista dei nodi aggiornati sia vuota (convergenza delle etichette ai costi dei cammini minimi da verso gli altri nodi).

*Indicazioni pratiche (per capire le cose)*

Nell’individuazione dei cammini minimi, ricostruisco la catena dei predecessori:

* Partendo solo nel disegno dall’ultimo nodo e poi vaglio tutte le alternativa prendendo gli ultimi nodi aggiornati e, tramite i predecessori, costruire un percorso minimo fino al nodo origine
  + Se ricostruendo i cammini minimi, torno su un arco già trovato in precedenza, in quel caso ho un ciclo negativo

Esempio di tabella e cosa scrivere per BF:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Iterazione | Nodo A | Nodo B | Nodo C | Nodo D | Nodo E | Nodo F | Aggiornamenti |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Dijkstra

*Indicazioni teoriche (da scrivere in esame)*

Legenda della tabella (la scrivo io per chiarezza di contenuto questa, ndr; occorre giustificare i passi):

* rappresenta l’etichetta minima di ogni iterazione
* rappresentano le etichette ancora da fissare
* Il segno \* rappresenta l’etichetta fissata
* Il segno – rappresenta l’etichetta controllata ma non aggiornata (normalmente ha stesso costo rispetto a quello che attualmente ha e quindi non vado ad aggiornare)
* Il segno rappresenta l’etichetta non controllata perché il nodo è già fissato
* Gli spazi vuoti servono per indicare che non considero più l’etichetta nelle varie iterazioni in quanto fissata
* L’algoritmo termina quando non ci sono più nodi in

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Iterazione | Nodo | Nodo | Nodo | Nodo | Nodo | Nodo |  |  |
|  |  |  |  |  |  |  |  |  |
|  | \* |  |  |  |  |  |  |  |
|  |  | - | \* |  |  |  |  |  |
|  |  | \* |  |  | - |  |  |  |
|  |  |  |  | - | \* | - |  |  |
|  |  |  |  | \* |  | - |  |  |
|  |  |  |  |  |  | \* |  |  |

Ad ogni iterazione, percorriamo il grafo e scegliamo il percorso con costo minore. Ad ogni iterazioni, scegliamo e fissiamo un’etichetta che ha costo minore, aggiungendo un’etichetta all’insieme di quelle fissate e controllando quanto accade per le altre (segnando etichette controllate ma non aggiornate oppure non controllate perché il nodo è già fissato).

Si procede inoltre con le verifiche della condizione di Bellman (vincolo duale) sugli archi che escono dall’etichetta minima e portano nodi nell’insieme dei nodi da fissare. Le etichette calcolate e i relativi puntatori rappresentano, rispettivamente, una soluzione duale ammissibile e i predecessori su dei cammini dall’origine ai diversi nodi.

L’algoritmo termina quando tutte le etichette sono state fissate, cioè quando tutti i nodi di vengono trasferiti in e quando tutte le etichette sono state correttamente fissate, quindi

AMPL

*Non appaiono quasi mai all’esame* e normalmente si tratta di tradurre le sommatorie in un file .mod; al massimo, può capitare di dover scrivere un file .dat.

Si segua questa struttura logica.

File.mod:

* Avere un *set* per indicare gli insiemi
* Avere dei *param* indicizzati con le graffe degli insiemi 🡪 es.
* Le variabili intere sono >=0 *integer*
* Le variabili reali sono >=0 e basta
* Le variabili comprese tra 0 ed 1 sono *binary*
* La f.o. viene scritta con *maximize/minimize* e poi nella forma
* Normalmente, i vincoli sono con *s.t.* e:
  + L’indice più esterno è quello per cui viene scritto il vincolo
    - Es. Indice più esterno è j e allora
* Attenzione a mettere (costante \* variabile)
  + Cioè scrivere

File .dat:

* Si descrivono gli insiemi con := su una sola linea
* Si descrivono eventuali parametri costanti con := in linea
* I param vanno con se in una sola linea, altrimenti
  + Se in forma matriciale (es diventa con in colonna e in riga; la forma di scrittura è
* Si aggiunge (tr) per mettere in riga e in colonna
* Ricordarsi di mettere i punti e virgola da tutte le parti, specie dopo le variabili matriciali

File .mod:

* reset
* model file.mod
* data file.dat
* option solver *unoqualunque*
* solve
* display funzioneobiettivo, variabile1, variabile2, etc.

Esempio notevole e di studio:

tImmagine che contiene testo

Descrizione generata automaticamente

*Immagine che contiene testo

Descrizione generata automaticamente*