Raccolta temi d'esame dell'insegnamento di Algoritmi Avanzati

Roberto Posenato

28 settembre 2010

Parte I

Anno accademico 2006/2007

1 Appello del 21/06/2007

Esercizio 1.1 (*max 7 punti*)**.** Si definisca cos'è un matroide e cosa si intende per *proprietà della sottostruttura ottima* dello stesso.

Esercizio 1.2 (*max 7 punti*)**.** Sia A un algoritmo con complessità in tempo pari a $O(2^n)$. Sia C un calcolatore in cui il tempo medio richiesto da un operazione in virgola mobile è $4^{-4}10^{-3}$ secondi. Un calcolatore C', dotato di un processore Intel Core 2 EX6800, ha una potenza di calcolo di 16000 MFLOPS. Qual è l'ordine di grandezza del tempo richiesto dai due calcolatori per risolvere A quando l'input ha dimensione 30. Fissato un tempo di computazione, qual è l'aumento della dimensione dell'istanza che può essere risolta usando C' rispetto C?

Esercizio 1.3 (*max 8 punti*)**.** Un impiegato ha n utenti in attesa di essere serviti. L'impiegato conosce in anticipo il tempo richiesto da ciascun utente: t_i per $i = 1, \ldots, n$.

Il tempo di gestione dell'utente i è $g_i = \sum_{j=1}^{i} t_j$, il tempo di attesa degli utenti prima i più il tempo richiesto da i per essere servito.

Si vuole minimizzare il tempo globale di gestione di tutti gli utenti $\sum_{i=1}^{n} g_i$.

Scrivere un algoritmo efficiente che determini lo scheduling ottimale degli utenti giustificando la tecnica usata per determinare l'algoritmo.

Esercizio 1.4 (*max 8 punti*)**.** Si deve intraprendere un lungo viaggio, che inizia idealmente dal punto 0. Lungo la strada ci sono n hotel, alle posizioni $a_1 < a_2 < a_3 < \ldots < a_n$, dove a_i è la distanza dalla partenza.

Si può guidare durante il giorno, mentre la notte ci si deve fermare in un hotel.

Idealmente si dovrebbero percorrere 200 km al giorno. Se si percorrono x km, la penalità che si deve pagare è $(200 - x)^2$.

Si illustri una soluzione algoritmica che determini l'elenco degli hotel in cui ci si deve fermare per minimizzare la somma totale delle penalità.

2 Appello del 10/07/2007

Esercizio 2.1 (*max 7 punti*)**.** Indicare cosa si intende per *principio di invarianza* nell'ambito dell'analisi degli algoritmi.

Esercizio 2.2 (*max 7 punti*)**.** Indicare cosa si intende per *indice di prestazione* di una soluzione e come questo concetto può essere applicato agli algoritmi.

Esercizio 2.3 (*max 8 punti*)**.** Un corriere deve consegnare ogni mattina n copie di un giornale in k edicole. Ciascuna edicola richiede $r_i(t)$ copie, dove i è l'edicola e t è il giorno.

La consegna di un lotto generico di copie all'edicola i costa $c_i(t)$.

Determinare una soluzione algoritmica che determini, giorno per giorno, quante copie consegnare a ciascuna edicola in modo che il costo globale sia minimo pur garantendo che ciascuna edicola abbia almeno una copia del giornale.

Esercizio 2.4 (*max 8 punti*)**.** Per un errore di sostituzione, un testo è stato privato di qualsiasi carattere di punteggiatura e spaziatura. Si vuole verificare se è possibile ricostruire il testo originale usando un dizionario, rappresentato dalla funzione `dict(w)` che restituisce vero se w è una parola, falso altrimenti.

Determinare un algoritmo che verifica se il testo può essere ricostruito come una sequenza di parole valide in tempo $O(n^2)$ dove n è la lunghezza del testo senza spaziatura/punteggiatura.

3 Appello del 04/09/2007

Esercizio 3.1 (*max 7 punti*)**.** Si consideri la seguente funzione per la risoluzione del problema dello ZAINO con ripetizione.

Funzione backZaino(p, v, i, r)

```
// p e v vettori dei pesi e dei valori degli elementi.
// i = tipo corrente, r = capacità residua.
1: b = 0;
2: for (k = i; k < p.length; k++) do // Prova ogni tipo k residuo
3:     if (p[k] ≤ r) then b = max(b, v[k] + backZaino(p, v, k, r − p[k]));
4: end for
5: return b;
```

Come si deve modificare affinché restituisca anche la soluzione ottima?

Esercizio 3.2 (*max 7 punti*)**.** Descrivere lo schema generale della tecnica del *Simulated annealing*.

Esercizio 3.3 (*max 8 punti*)**.** Ci sono n oggetti da ordinare usando le due relazioni "$<$" e "$=$". Se, per esempio, si hanno gli oggetti a, b, c, i possibili ordinamenti sono:

$$\begin{array}{lllll} a=b=c & a=b<c & a<b=c & a<b<c & a<c<b \\ a=c<b & b<a=c & b<a<c & b<c<a & b=c<a \\ c<a=b & c<a<b & c<b<a & & \end{array}$$

Determinare un algoritmo che calcoli, in funzione di n, il *numero* dei possibili ordinamenti diversi. L'algoritmo deve lavorare in tempo $O(n^2)$ e in spazio $O(n)$.

Esercizio 3.4 (*max 8 punti*)**.** Dati n programmi con tempi di esecuzione $t_1, t_2, \ldots, t_n$, si vuole determinare un ordinamento degli stessi in modo da minimizzare il tempo medio di completamento $1/n \sum_i c_i$ dove c_i è la somma dei tempi di esecuzione del programma i e di tutti i programmi che lo precedono nell'ordinamento finale.

Si progetti un algoritmo di ricerca locale per risolvere il problema.

4 Appello del 18/09/2007

Esercizio 4.1 (*max 7 punti*)**.** Si definisca cosa si intende per problema r-approssimabile.
Due studiosi sostengono che il problema MINIMO RICOPRIMENTO DI VERTICI è, rispettivamente, 2-approssimabile e ½-approssimabile. Chi ha ragione? Perché?

Esercizio 4.2 (*max 7 punti*)**.** Indicare cosa si intende per strategia *hill climbing* e dove viene solitamente applicata.

Esercizio 4.3 (*max 8 punti*)**.** All'interno di un'azienda si vogliono organizzare dei pranzi di lavoro per migliorare la socializzazione dei dipendendi. Il personale è organizzato in modo gerarchico e il livello massimo è costituito dall'amministratore delegato.

I dipendenti che partecipano ad un pranzo non devono essere in relazione gerarchica né diretta né indiretta.

Data una lista di k dipendenti liberi per il primo pranzo, si determini un algoritmo che calcoli in modo efficiente il sottoinsieme massimo di questi k dipendenti che possono partecipare rispettando il vincolo dato.

Dare la prova di correttezza e la complessità in tempo dell'algoritmo proposto.

Esercizio 4.4 (*max 8 punti*)**.** Ci sono n oggetti da ordinare usando le due relazioni "$<$" e "$=$". Se, per esempio, si hanno gli oggetti a, b, c, i possibili ordinamenti sono:

$$\begin{array}{lllll} a=b=c & a=b<c & a<b=c & a<b<c & a<c<b \\ a=c<b & b<a=c & b<a<c & b<c<a & b=c<a \\ c<a=b & c<a<b & c<b<a & & \end{array}$$

Determinare un algoritmo che calcoli, in funzione di n, il *numero* dei possibili ordinamenti diversi. L'algoritmo deve lavorare in tempo $O(n^2)$ e in spazio $O(n)$.

Parte II
Anno accademico 2007/2008

5 Appello del 25/06/2008

Esercizio 5.1 (*max 7 punti*)**.** Si definisca cosa si intende per *tecnica memoization* e si indichi quali sono i principali vantaggi e svantaggi di questa tecnica.

Esercizio 5.2 (*max 7 punti*)**.** Si definisca cosa si intende per *Tabu search* e si dia, in modo schematico, la descrizione dei tipi di proibizioni noti.

Esercizio 5.3 (*max 8 punti*)**.** Data una moneta non bilanciata in cui la probabilità che esca testa in un lancio è il valore p non noto e i lanci non si influiscono, si costruisca un semplice algoritmo che generi sequenze bilanciate di bit casuali usando tale moneta.

Esercizio 5.4 (*max 8 punti*)**.** È noto che è opportuno mantenere un albero binario di ricerca bilanciato per minimizzare la media dei confronti necessari per ricercare un elemento. Tale fatto è vero se si assume che gli elementi dell'albero hanno pari probabilità di essere ricercati.

Se gli elementi di un albero di ricerca hanno però una probabilità $p_i (i = 1, \ldots, n)$ di essere ricercati (assumiamo che $\sum_{i=1}^{n} p_i = 1$), il bilanciamento non è la scelta migliore. Un albero di ricerca in cui le chiavi sono sistemate in modo opportuno secondo la loro probabilità di ricerca determina un numero medio di confronti necessario minore.

Per fissare un po' le basi, se d_i è la profondità del nodo i (ricordare che $d_{\text{root}} = 0$), allora il numero di confronti per raggiungere il nodo i è $d_i + 1$ e, dato un albero di ricerca in cui le chiavi sono distribuite secondo $p_i (i = 1, \ldots, n)$, il numero di confronti medio è $C = \sum_{i=1}^{n} p_i (d_i + 1)$.

Si determini un algoritmo che, dati in input il vettore K delle chiavi e il vettore P della probabilità di ricerca per ciascuna di esse, restituisca il valore del numero di confronti medio dell'albero di ricerca ottimo.

Suggerimento: Si consideri $C_{i,j}$ = numero medio di confronti necessari quando il sottoalbero è formato da $k_i, \ldots, k_j$ e che il numero di confronti medio per giungere a $C_{i,j}$ è la probabilità dell'albero.

6 Appello del 22/07/2008

Esercizio 6.1 (*max 7 punti*)**.** Si descriva quali sono le fasi in cui si può suddividere il paradigma *divide et impera* in generale e quale dovrebbe essere la loro complessità computazionale affinché l'algoritmo sia efficiente.

Esercizio 6.2 (*max 7 punti*)**.** Si definisca cosa si intende per *vettore k-promettente* e si dia il pseudo-codice di un paradigma/tecnica che usa questa struttura dati.

Esercizio 6.3 (*max 8 punti*)**.** Si scriva un algoritmo dettagliato basato su un'appropriata tecnica che stampi tutte le permutazioni di n caratteri di una stringa S.

Esercizio 6.4 (*max 8 punti*)**.** Si consideri il seguente problema:

PROBLEMA MINIMO RICOPRIMENTO DI INSIEME (MINIMUM SET COVER)

DESCRIZIONE: Una famiglia di insiemi $F = \{S_1, S_2, \ldots, S_n\}$, ciascuno sottoinsieme di un insieme finito U.

QUESITO: Determinare una sotto famiglia $C \subseteq F$, detta *cover*, tale che l'unione di tutti i suoi insiemi è pari a U. In altre parole $\bigcup_{S_i \in C} S_i = U$.

MISURA: $|C|$

È dimostrato che la versione di decisione del problema è NP-completa. Si determini un algoritmo di risoluzione approssimato polinomiale in tempo. Si valuti l'indice di prestazione della soluzione determinata dall'algoritmo per l'istanza in cui U ha $n = 2^{k+1} - 2$ elementi e $F = \{S_1, \ldots, S_k, T_0, T_1\}$ è costituito da k sottoinsiemi disgiunti di dimensione 2^i dove i è l'indice dell'insieme e dagli insiemi T_0 e T_1 che sono disgiunti tra loro e che contengono, ciascuno, metà elementi di ciascun insieme S_i.

Parte III
Anno accademico 2008/2009

7 Appello del 02/04/2009

Esercizio 7.1 (*max 7 punti*)**.** Si descriva in modo sintetico come impostare una soluzione di programmazione dinamica per un problema di ottimizzazione.

Se una soluzione al passo i, j, k-esimo richiede di determinare soluzioni in posizioni $1 \dots i-1, j-2 \dots j-1, 1 \dots k-1$, qual è l'ordine di grandezza della quantità di memoria necessaria per determinare la soluzione? La soluzione è data dalla soluzione della chiamata (n, n, n).

Esercizio 7.2 (*max 7 punti*)**.** Si dimostri come una soluzione parziale minima di un matroide pesato può essere completata ottenendo una soluzione ottima del problema assumendo che le soluzioni ammissibili sono quelle rappresentare da insiemi di indipendenza massimali

Si dia un esempio di problema computazionale che può essere modellato come un problema di matroide pesato.

Esercizio 7.3 (*max 8 punti*)**.** Si descriva un algoritmo che garantisca che il risultato di un algoritmo Monte Carlo sia corretto con confidenza del 99.5% sapendo che l'algoritmo è $\frac{1}{2} + \epsilon$-corretto. Si ricorda che la tabella della distribuzione di probabilità normale indica che $\Pr[X < \mu(X) - 2.576\sigma(X)] \approx 0.005$.

Esercizio 7.4 (*max 8 punti*)**.** Dato un vettore A, si dice che ammette elemento *majority* se più della metà degli elementi sono uguali. Gli elementi dell'array non sono confrontabili con operatori diversi da ==. Usando una delle tecniche viste durante il corso, determinare un algoritmo con complessità $O(n)$ che determina l'elemento *majority* (se esite). *Suggerimento: Si accoppiano casualmente gli elementi: $n/2$ coppie. Si tengono solo le coppie con elementi uguali. Gli elementi diventano $n/2$ al più. Se hanno un majority...*

8 Appello del 09/09/2009

Esercizio 8.1 (*max 7 punti*)**.** Si descriva come si può impostare una soluzione Simulated Annealing per il problema MAX-SAT (Massima soddisfacibilità di formule booleane in forma normale congiunta).

Esercizio 8.2 (*max 7 punti*)**.** Un multi-insieme può contenere più volte uno stesso elemento. Un multi-insieme di n elementi può quindi ammettere meno di $n!$ permutazioni distinte. Si analizzi il problema di produrre tutte le permutazioni distinte di un multi-insieme e si proponga un algoritmo risolutore efficiente indicando quale tecnica algoritmica si è adottata e qual è la complessità dell'algoritmo.

Esercizio 8.3 (*max 8 punti*)**.** Si consideri il problema MAX-SAT. Si proponga un'euristica in grado di determinare una soluzione approssimata il cui errore relativo sia al massimo 1/2. Indicare quale tecnica si è adottata e qual è la complessità dell'algoritmo (che deve essere polinomiale al più!).

Esercizio 8.4 (*max 8 punti*)**.** Dato un array di n numeri reali, si consideri il problema di determinare qual è il sottovettore di elementi continui la cui somma è massima.

Si proponga un algoritmo che determina il valore della somma in un tempo $\Theta(n)$ indicando quale tecnica si è adottata.

Parte IV
Anno accademico 2009/2010

9 Prova di appello del 13/01/2010

Esercizio 9.1 (*max 7 punti*)**.** Indicare cosa si intende per *principio di invarianza* nell'ambito dell'analisi degli algoritmi e qual è la sua conseguenza immediata.

In complessità il *principio di invarianza* può essere visto come un teorema se si considera uno specifico modello di calcolo. Enunciare il teorema e indicare perché è equivalente al principio di invarianza.

Esercizio 9.2 (*max 7 punti*)**.** Nell'ambito della teoria degli algoritmi, si definisca cosa si intende per *tecnica memoization* e si indichi quali sono i principali vantaggi e svantaggi di questa tecnica.

Nell'ambito della complessità, si definisca la classe $\mathsf{TIME}(f(n))$. Se $x \in \mathsf{TIME}(f(n))$, in quali altre classi $\mathsf{TIME}(g(n))$ è presente? Giustificare la risposta.

Esercizio 9.3 (*max 8 punti*)**.** Si consideri il problema del MINIMO RICOPRIMENTO DI VERTICI.

- Si determini una funzione upper bound e una lower bound per le soluzioni parziali pensando di dover sviluppare un algoritmo branch & bound.
- Si determini inoltre la sua complessità computazionale.

Esercizio 9.4 (*max 8 punti*)**.** Si dimostri che il problema del commesso viaggiatore è NP-completo esibendo una riduzione logaritmica in spazio. Si dia quindi un algoritmo diverso dalla ricerca esaustiva semplice che determini una soluzione ottima.

10 Prova di appello del 20/01/2010

Esercizio 10.1 (*max 7 punti*)**.** Parte complessità:
Descrivere la differenza tra la versione di decisione e la versione costruttiva di un problema di ottimizzazione. È sempre possibile, dato un algoritmo polinomiale in tempo per la versione di decisione, determinare un algoritmo polinomiale in tempo per la versione costruttiva? Giustificare la risposta.

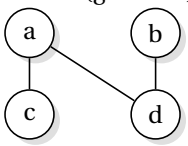
Parte algoritmi avanzati:
Presentare un algoritmo polinomiale in tempo per la versione di decisione di un problema a scelta e, quindi, modificare tale algoritmo per determinare, sempre in tempo polinomiale, la soluzione alla versione costruttiva. *(max 7 punti)*

Esercizio 10.2 (*max 7 punti*)**.** Parte complessità:
Si definisca cosa si intende per *funzione propria* e se ne dia un esempio con dimostrazione.

Parte algoritmi:
Si definisca cos'è un matroide e cosa si intende per *proprietà della sottostruttura ottima* dello stesso.

Esercizio 10.3 (*max 16 punti*)**.** Parte di complessità:
Si dimostri che il problema MASSIMO INSIEME DI INDIPENDENZA (D) è in P quando il grado del grafo è al più 2. Si determini l'ordine di grandezza dell'algoritmo polinomiale sufficiente per dimostrare l'appartenenza.

Parte di algoritmi:
Si proponga uno schema di algoritmo che determini la soluzione esatta del problema MASSIMO INSIEME DI INDIPENDENZA (generale) adottando la tecnica del backtracking. Si disegni l'albero implicito visitato dall'algoritmo per il grafo



11 Appello del 02/02/2010

Esercizio 11.1 (max 7 punti per parte)**.**
Parte Complessità
Si dia la definizione formale della classe di complessità EXP e si dimostri la relazione esistente tra EXP e la classe PSPACE.
Parte Algoritmi Si definisca cosa si intende per *tecnica memoization* e si indichi quali sono i principali vantaggi e svantaggi di questa tecnica.

Esercizio 11.2 (max 7 punti per parte)**.**
Parte Complessità
Si dimostri in modo esplicito che 2-COLORABILITÀ è in P.
Parte Algoritmi Si descriva come si può impostare una soluzione Simulated Annealing per il problema MAX-SAT (Massima soddisfacibilità di formule booleane in forma normale congiunta).

Esercizio 11.3 (max 16 punti per parte)**.**
Parte Complessità
Dare una dimostrazione che MASSIMA CRICCA(D) è NP-completo senza ricorrere alla completezza di MASSIMO INSIEME DI INDIPENDENZA(D) o di MINIMO RICOPRIMENTO DI VERTICI(D).
Parte Algoritmi
Determinare un algoritmo esatto per MASSIMA CRICCA basato sul branch & bound scrivendo in modo esplicito le due funzioni bound che l'algoritmo usa. Suggerimento: una soluzione anche non ottima del problema COLORAZIONE è un upper bound.

12 Appello del 16/02/2010

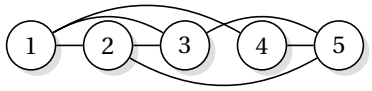
Esercizio 12.1 (max 8 punti per parte)**.**
Parte Complessità
Si dia l'enunciato del teorema di Savitch e si indichi qual è la conseguenza sulla classe NSPACE($f(n)$). Se $f(n)$ è una funzione propria di complessità $O(\log n)$, qual è la conseguenza sulla classe NSPACE($f(n)$).
Parte Algoritmi Si indichino le fasi con cui si dovrebbe sviluppare un algoritmo di programmazione dinamica.

Esercizio 12.2 (max 8 punti per parte)**.**
Parte Complessità
Dare la definizione di riduzione polinomiale in tempo ($\preceq_m$) e dimostrare che è una relazione riflessiva. *Suggerimento: è come la riduzione logaritmica in spazio ma pone un vincolo sul tempo anziché sullo spazio.*
Parte Algoritmi Si consideri il problema COLORABILITÀ(Dato un grafo connesso di n nodi, colorare i nodi con il minor numero possibile di colori in modo che nessuna coppia di nodi adiacenti abbiano lo stesso colore). Si consideri quindi il seguente algoritmo:

Procedura COL(G)

```
   // G = (V, E) è un grafo di n nodi
 1: sol = [0, ..., 0];
 2: for (i = 1; i ≤ n; i++) do
 3:     C = {c_1, c_2, c_3, ..., c_n};
 4:     for (j = 1; j ≤ i − 1; j++) do
 5:         if ({i, j} ∈ E) then
 6:             C = C \ {sol[j]};
 7:         end if
 8:     end for
 9:     sol[i] = min{c | c ∈ C};
10: end for
11: return sol
```

Scrivere la colorazione che si ottiene per il grafo



Risolve il problema? In caso affermativo dare una dimostrazione di correttezza, altrimenti esibire un controesempio.

Esercizio 12.3 (max 16 punti per parte)**.**
Parte Complessità
Dare una dimostrazione che MINIMO RICOPRIMENTO DI VERTICI(D) è NP-completo senza ricorrere alla completezza di MASSIMO INSIEME DI INDIPENDENZA(D) o di MASSIMA CRICCA(D).
Parte Algoritmi
Determinare un algoritmo esatto per la versione di valutazione del problema k-SOTTOSEQUENZA di interi. Un k-sottosequenza di X è una sottosequenza di X in cui compaiono al più k elementi consecutivi di X. Il problema chiede, data una sequenza X di n interi e un intero $k < n$, di determinare una k-sottosequenza di X con somma degli elementi massima possibile. Per esempio, la sequenza $X = [4,6,-1,4,7,5,4,7]$ ammette la 3-sottosequenza di valore 33: $[4,6,4,7,5,7]$.
Suggerimento: caratterizzare la versione di valutazione in modo ricorsivo usando due parametri...

13 Appello del 24/06/2010

Esercizio 13.1 (max 8 punti per parte)**.**
Parte Complessità
Dimostrare che NTIME(n) $\subset$ PSPACE. Se fosse $\bigcup_{k>0}$ NTIME(n^k) cosa cambierebbe?
Parte Algoritmi Si definisca cos'è un matroide e cosa si intende per *proprietà della sottostruttura ottima* dello stesso.

Esercizio 13.2 (max 8 punti per parte)**.**
Parte Complessità
Descrivere l'errore nella seguente dimostrazione *errata* che P $\neq$ NP. *Si consideri un algoritmo per* SODDISFACIBILITÀ *del tipo: "Su input ϕ, prova tutti i possibili assegnamenti alle variabili. Accetta se almeno un assegnamento soddisfa ϕ". Questo algoritmo chiaramente richiede tempo esponenziale. Perciò* SODDISFACIBILITÀ *ha complessità esponenziale, quindi* SODDISFACIBILITÀ $\notin$ P. *Dato che* SODDISFACIBILITÀ *è in* NP*, si ottiene che* P $\neq$ NP.
Parte Algoritmi Descrivere lo schema generale della tecnica del *Simulated annealing.*

Esercizio 13.3 (max 16 punti per parte)**.**
Parte Complessità
Si consideri il problema del MINIMO RICOPRIMENTO DI VERTICI(D) su grafi privi di triangoli. Si dimostri che il problema è NP-completo.
Parte Algoritmi
Progettare un algoritmo di tipo *divide et impera* che, dato un vettore di n interi, trovi un sottovettore (sequenza di elementi consecutivi) con somma dei suoi elementi massima. La complessità deve essere $O(n \log n)$. *Suggerimento: ricordare che il vettore con somma massima può essere a cavallo del punto di divisione.*

14 Appello del 15/07/2010

Esercizio 14.1 (max 8 punti per parte)**.**
Parte Complessità
Dare le definizioni delle classi NP e EXP. Dare un esempio di problema che sta in EXP e che non è stato ancora dimostrato essere in NP.
Parte Algoritmi Cosa si intende per "algoritmo di Monte Carlo".

Esercizio 14.2 (max 8 punti per parte)**.**
Parte Complessità
Si enunci il teorema di gerarchia in spazio e si dimostri che conseguenza ha sulla catena di relazioni tra le principali classi di complessità.
Parte Algoritmi Si assuma che il problema di massimizzazione Π sia 3/2-approssimabile. Cosa significa? Data un'istanza I di Π, in quale intervallo (rispetto alla soluzione ottima $m^*(I)$) cade una soluzione approssimata per Π se soddisfa la proprietà di approssimazione del problema stesso?

Esercizio 14.3 (max 16 punti per parte)**.**
Parte Complessità
Si consideri il problema 3-SODDISFACIBILITÀ in cui in ogni clausola può avere dimensione < 3 e, rispetto a tutte le clausole, ogni variabile compare al più 3 volte e ciascun letterale al più 2. Dimostrare che è NP-completo.
Parte Algoritmi
Progettare un algoritmo di tipo programmazione dinamica che, dato una matrice M di $n \times m$ interi, determina il vettore di somma minima $[m_{1,j_1}, m_{2,j_2}, \ldots, m_{n,j_n}]$ dove $1 \leq j_1 \leq \ldots \leq j_n \leq m$. Il tempo di computazione deve essere $O(nm^2)$.

15 Appello del 14/09/2010

Esercizio 15.1 (max 8 punti per parte)**.**
Parte Complessità
Si enunci il teorema di Savitch e si illustri brevemente qual è la conseguenza principale di tale teorema.
Parte Algoritmi
Si definisca cosa si intende per problema r-approssimabile.
Due studiosi sostengono che il problema MIN EDGE COLORING è, rispettivamente, $\frac{4}{3}$-approssimabile e $\frac{1}{4}$-approssimabile. Chi ha ragione? Perché?

Esercizio 15.2 (max 8 punti per parte)**.**
Parte Complessità
Si enunci il teorema di gerarchia in tempo e si dimostri che conseguenza ha sulla catena di relazioni tra le principali classi di complessità.
Parte Algoritmi Si consideri il problema MAX-SAT. Si proponga un'euristica in grado di determinare una soluzione approssimata il cui errore relativo sia al massimo 1/2. Indicare quale tecnica si è adottata e qual è la complessità dell'algoritmo (che deve essere polinomiale al più!).

Esercizio 15.3 (max 16 punti per parte)**.**
Parte Complessità
Il problema del ricoprimento con cricche chiede di verificare se dato un grafo finito e un intero d è possibile partizionare il grafo in d sottografi tali che ciascuno di essi è una cricca. Si dimostri che il problema è NP-completo.
Suggerimento: problema k-colorabilità.
Parte Algoritmi
Alcuni studenti del Corso di Studi in Informatica pensano di essere in grado di laurearsi facendo il minimo sforzo in termini studio. Sia $E = \{e_1, e_2, \ldots, e_n\}$ l'insieme degli n esami attivati nel Corso di Studi, dove per ogni $i = 1, 2, \ldots, n$, l'esame e_i vale c_i crediti formativi. Si assuma che ogni esame e_i abbia un coefficiente d_i che rappresenta il grado di difficoltà dell'esame stesso. Gli studenti possono redigere il loro piano di studio individuale scegliendo nella lista degli

esami attivati un insieme di esami tali che la somma dei crediti corrispondenti sia almeno P. Progettare un algoritmo che redige un piano di studio regolare con difficoltà minima in $O(nP)$.

16 Appello del 28/09/2010

Esercizio 16.1 (max 8 punti per parte)**.**
Parte Complessità
Quali sono i risultati principali che permettono di dimostrare che NL $\subset$ PSPACE. Giustificare la risposta.
Parte Algoritmi
Si definisca cosa si intende per *Tabu search* e si dia, in modo schematico, la descrizione dei tipi di proibizioni noti.

Esercizio 16.2 (max 8 punti per parte)**.**
Parte Complessità
Indicare quali delle seguenti relazioni sono errate, giustificando la risposta:

1. SODDISFACIBILITÀ $\leq_{\log}$ CIRCUIT SAT;
2. CIRCUIT VALUE $\leq_{\log} \emptyset$;
3. MASSIMO FLUSSO $\leq_{\log}$ RAGGIUNGIBILITÀ.

Parte Algoritmi Si consideri il problema MAX-SAT. Si proponga un'euristica in grado di determinare una soluzione approssimata il cui errore relativo sia al massimo $1/2$. Indicare quale tecnica si è adottata e qual è la complessità dell'algoritmo (che deve essere polinomiale al più!).

Esercizio 16.3 (max 16 punti per parte)**.**
Parte Complessità
Dimostrare che se esiste un problema P tale che sia P che P^c sono NP-completi, allora NP = coNP.
Si ricorda P^c è il problema complemento e coNP è la classe dei problemi complemento dei problemi in NP.
Parte Algoritmi
In un grafo orientato G un sottoinsieme A degli archi è detto univoco se non contiene 2 o più archi uscenti dallo stesso nodo. Descrivere un algoritmo efficiente che preso in input un grafo G orientato con pesi positivi sugli archi, trovi un insieme univoco A di archi di G di peso **massimo**. Provare la correttezza dell'algoritmo proposto e valutare la complessità.