UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"

Corso di Laurea Triennale in Matematica

Algoritmi di approssimazione per la massimizzazione di funzioni submodulari

Relatore
Prof. Marco Di Summa

Laureando: Giovanni Gaio
Matricola: 2052828

Anno Accademico 2023/2024

19 luglio 2024

Indice

Introduzione

Le funzioni submodulari sono una classe di funzioni $f : \mathcal{P}(X) \to \mathbb{R}$, con X insieme finito, caratterizzate dalla semplice proprietà dell'avere rendimenti decrescenti (*diminishing returns*). In particolare se si prende un elemento $x \in X$ e due insiemi $A \subseteq B \subseteq X \setminus \{x\}$ allora per una funzione f submodulare vale che $f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$, cioè l'aggiunta di x fa aumentare meno il valore di f quando l'insieme è più grande.

Questa proprietà è molto naturale in moltissimi contesti pratici: alcune funzioni submodulari nascono da problemi concreti, come nel caso del problema di *facility location* in cui si vuole massimizzare il guadagno dato dalle posizioni di filiali in diverse città [9, 1]. Oppure in ambiti economici la submodularità è una proprietà comune in molti contesti, ad esempio come proprietà di alcune funzioni di utilità [1].

L'interesse per questo tipo di funzioni è dato anche dai numerosi problemi che si riconducono ad esse, tra questi ci sono: i problemi di taglio sui grafi e le coperture di insiemi; altri esempi nascono da problemi nell'ambito dell'intelligenza artificiale [2], come il problema di *feature selection*, cioè il problema di scelta delle variabili da usare nel modello di apprendimento automatico.

Dalla proprietà che definisce le funzioni submodulari si possono costruire algoritmi per minimizzarle in tempo polinomiale [5]. Lo stesso non è vero per la massimizzazione. Più precisamente data una funzione submodulare f si vuole trovare

$$S \in C \text{ tale che } f(S) = \max_{S \in C} f(A)$$

dove l'insieme $C \subseteq \mathcal{P}(X)$ è definito solitamente ponendo alcuni vincoli su $\mathcal{P}(X)$. Trovare esattamente il valore massimo su tutto $\mathcal{P}(X)$ di una funzione submodulare è un problema NP-difficile [7].

Nel presente lavoro si vedranno alcuni algoritmi che trovano degli insiemi $S \in C \subseteq \mathcal{P}(X)$ tali che $f(S) \geq \alpha \left(\max_{A \in C} f(A)\right)$, per un certo $\alpha \in (0, 1)$ che dipende soltanto dall'algoritmo. Questo vuol dire che il valore di $f(S)$ è abbastanza vicino al massimo possibile.

Il caso più semplice di massimizzazione di una funzione submodulare senza vincoli, cioè con $C = \mathcal{P}(X)$, sarà trattato nel Capitolo 2. Saranno presentati alcuni algoritmi che approssimano il valore cercato. Si vedranno prima degli algoritmi meno precisi ma più semplici per poi presentare un algoritmo più sofisticato, che garantisce un fattore di approssimazione di $\frac{1}{2}$. Questo vuol dire che si ha la garanzia di trovare un insieme S tale per cui $f(S) \geq \frac{1}{2} \left(\max_{S \in \mathcal{P}(X)} f(A)\right)$.

Poi, nel Capitolo 3, ci si preoccuperà di verificare che, sotto determinate condizioni, l'algoritmo 3 presentato in questo lavoro è il migliore che si può raggiungere. In particolare si richiede che la funzione f non sia valutata un numero di volte che cresce esponenzialmente rispetto alla dimensione dell'insieme X.

A questo punto il problema di massimizzazione di una funzione submodulare senza vincoli si può considerare risolto, dato che si conosce un algoritmo che garantisce di fornire la migliore approssimazione che si può trovare in tempo polinomiale.

In moltissime applicazioni reali però è utile limitare la ricerca del massimo di una funzione submodulare imponendo un qualche tipo di vincolo. Ci sono diversi tipi di vincolo che sono stati studiati, tra questi i più importanti sono forse i vincoli di matroide e *knapsack* [10]. Il più semplice esempio è però quello di porre un limite alla cardinalità degli insiemi, e sarà infatti questo il tema del Capitolo 4.

Però prima di tutto ciò è opportuno definire formalmente le funzioni submodulari, capirne le proprietà fondamentali e individuarne alcuni esempi. Questo verrà fatto nel prossimo capitolo.

Capitolo 1

Definizioni di base

Omnium rerum principia parva sunt.

Marco Tullio Cicerone

Per tutta la trattazione indicheremo con X un insieme finito di cardinalità n.

1.1 Funzioni submodulari

1.1.1 Definizione e proprietà di base

Definizione 1.1. Una funzione $f : \mathcal{P}(X) \to \mathbb{R}$ si dice **submodulare** [3, 7, 10] se $\forall S, T \subseteq X$ vale:

$$f(S \cup T) + f(S \cap T) \leq f(S) + f(T).$$

Le funzioni submodulari si caratterizzano facilmente tramite le loro derivate discrete, cioè la variazione del valore della funzione all'aggiunta di un elemento.

Proposizione 1.2 (Caratterizzazione delle funzioni submodulari)**.** *Una funzione $f : \mathcal{P}(X) \to \mathbb{R}$ è submodulare se e solo se $\forall A \subseteq B \subset X$, e $x \in X \setminus B$, allora:*

$$f\left(B \cup \{x\}\right) - f\left(B\right) \leq f\left(A \cup \{x\}\right) - f\left(A\right).$$

Dimostrazione. ($\Longrightarrow$) è sufficiente prendere $S = B$ e $T = A \cup \{x\}$.
($\Longleftarrow$) Siano $S, T \subseteq X$ qualunque. Allora definiamo l'insieme $\{x_1, \ldots, x_k\} = T \setminus S$, cioè $S \cup T = S \cup \{x_1, \ldots, x_k\}$ quindi, usando l'ipotesi:

$$\begin{aligned} f(S \cup T) - f(S) &= \sum_{i=1}^{k} f(S \cup \{x_1, \ldots, x_i\}) - f(S \cup \{x_1, \ldots, x_{i-1}\}) \\ &\leq \sum_{i=1}^{k} f((S \cap T) \cup \{x_1, \ldots, x_i\}) - f((S \cap T) \cup \{x_1, \ldots, x_{i-1}\}) \\ &= f(T) - f(S \cap T). \end{aligned}$$

Per il caso $i = 1$ nella somma si è supposto di avere $\{x_1, \ldots, x_{i-1}\} = \emptyset$. Le somme sono entrambe delle somme telescopiche, che giustificano quindi le relative uguaglianze. □

Questa proposizione mostra che le funzioni submodulari sono caratterizzate equivalentemente dall'avere derivata discreta decrescente, in modo analogo alle funzioni concave nel caso continuo. Queste però possono essere massimizzate esattamente in tempo polinomiale, mentre come si vedrà nel Capitolo 3 questo non è vero per funzioni submodulari.

1.1.2 Esempi di funzioni submodulari

Due classi di funzioni submodulari sono notevoli:

- le **funzioni submodulari simmetriche**: le $f : \mathcal{P}(X) \to \mathbb{R}$ submodulari t.c.
$$f(S) = f(X \setminus S) \quad \forall S \subseteq X;$$
- le **funzioni submodulari monotone**: le $f : \mathcal{P}(X) \to \mathbb{R}$ submodulari t.c.
$$\forall S \subseteq T \subseteq X \implies f(S) \leq f(T).$$
La monotonia è equivalente all'avere le derivate discrete sempre positive.

  Nota: la monotonicità è soltanto crescente. Il caso submodulare decrescente si riduce a quello crescente in modo semplice passando ai complementari. Per questo non verrà considerato.

Alcuni esempi di funzioni submodulari sono i seguenti:

1. *funzioni modulari:* cioè le funzioni $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ per cui $\exists w : X \to \mathbb{R}$ t.c. $f(S) = \sum_{x \in S} w(x)$.

   Come suggerisce il nome queste funzioni soddisfano con l'uguaglianza la relazione che definisce la submodularità. Infatti per queste funzioni la derivata discreta $f(S \cup \{x\}) - f(S)$ è pari a $w(x)$, indipendentemente da $S \subseteq X \setminus \{x\}$ per tutti gli $x \in X$.

2. *Funzioni di taglio su un grafo diretto* $G = (V, E)$, con pesi non negativi sugli archi $w : E \to \mathbb{R}_{\geq 0}$ allora la funzione $f : \mathcal{P}(V) \to \mathbb{R}$ di taglio è $f(S) = \sum_{(u,v) \in E : u \in S, v \notin S} w((u, v))$.

   Queste sono funzioni submodulari: infatti per qualunque $A \subseteq B \subseteq V$, $u \in V \setminus B$, le derivate discrete in $\{u\}$ sono decrescenti:
$$\begin{aligned} f(A \cup \{u\}) - f(A) &= \sum_{(u,v) \in E : v \notin A} w((u, v)) - \sum_{(v,u) \in E : v \in A} w((v, u)) \\ &\geq \sum_{(u,v) \in E : v \notin B} w((u, v)) - \sum_{(v,u) \in E : v \in B} w((v, u)) \\ &= f(B \cup \{u\}) - f(B). \end{aligned}$$

Questa disuguaglianza è vera perché: $\sum_{(u,v)\in E:v\notin A} w((u,v)) \geq \sum_{(u,v)\in E:v\notin B} w((u,v))$, data l'inclusione $X \setminus A \supseteq X \setminus B$; mentre l'inclusione inversa vale per gli insiemi complementari $A \subseteq B$, ciò implica $\sum_{(v,u)\in E:v\in A} w((v,u)) \leq \sum_{(v,u)\in E:v\in B} w((v,u))$. Per entrambe queste disuguaglianze è necessario che i pesi $w((u,v))$ siano non negativi.

Si noti che le funzioni in questa classe in generale non sono monotone, infatti $f(\emptyset) = f(V)$. Inoltre nel caso in cui G sia un grafo non diretto, la funzione di taglio è submodulare simmetrica.

3. *Facility location:* si supponga di avere un insieme X di possibili posizioni in cui costruire filiali, e un insieme di città C a cui fornire un servizio. Allora l'utilità data da un insieme di filiali $S \subseteq X$, in alcune situazioni, si può stimare con la funzione:

$$f(S) = \sum_{i\in C} \max_{s\in S} a_{i,s}$$

dove $a_{i,j} \in \mathbb{R}$ indica l'utilità che la filiale j dà alla città i. Questa espressione è motivata dal fatto che ci si può ragionevolmente aspettare che i cittadini di ogni città si serviranno nella filiale migliore per loro.

È facile allora vedere che le funzioni così costruite sono submodulari monotone: consideriamo le derivate discrete su due insiemi $A \subseteq B \subset X$ e $x \in X \setminus B$:

$$\begin{aligned} f(B\cup\{x\}) - f(B) &= \sum_{i\in C} \left(\max_{s\in B\cup\{x\}} a_{i,s} - \max_{s\in B} a_{i,s} \right) = \sum_{i\in C} \max\left\{0, a_{i,x} - \max_{s\in B} a_{i,s}\right\} \\ &\leq \sum_{i\in C} \max\left\{0, a_{i,x} - \max_{s\in A} a_{i,s}\right\} = f(A\cup\{x\}) - f(A). \end{aligned}$$

Esistono e sono studiate numerose varianti di questo problema, e quella presentata è solo una delle versioni più semplici.

Altri esempi di funzioni submodulari derivano da problemi come *feature selection* [2], la copertura pesata di un insieme [2, 6], l'entropia [10], l'informazione mutua [10], e molti altri. Infatti la proprietà di avere incrementi decrescenti della submodularità è una proprietà che si riscontra in moltissime situazioni naturali.

1.1.3 Operazioni che rispettano la submodularità

Date $f, g : \mathcal{P}(X) \to \mathbb{R}$ submodulari, allora:

1. $\forall c \in \mathbb{R}$ $f + c$ è submodulare.

2. $\forall c \in \mathbb{R}_{>0}$ $c \cdot f$ è submodulare.

3. $\forall S \subseteq X$, $f_S : \mathcal{P}(X) \to \mathbb{R}$, $f_S : A \mapsto f(A \cup S)$ è funzione submodulare, infatti consideriamo le derivate discrete rispetto a $x \in X$ sugli insiemi $A \subseteq B \subseteq X$:

$$\begin{aligned} f_S(B \cup \{x\}) - f_S(B) &= f(B \cup S \cup \{x\}) - f(B \cup S) \\ &\leq f(A \cup S \cup \{x\}) - f(A \cup S) = f_S(A \cup \{x\}) - f_S(A). \end{aligned}$$

4. $\hat{f} : \mathcal{P}(X) \to \mathbb{R}$, $\hat{f} : A \mapsto f(X \setminus A)$ è funzione submodulare, infatti la definizione è simmetrica rispetto al complemento.

5. La funzione $\min(f, c)$ è submodulare $\forall c \in \mathbb{R}$ nel caso in cui f sia anche monotona. Infatti sia $A \subseteq B \subset X$, e $x \in X \setminus B$:

$$\min\{f(A \cup \{x\}), c\} - \min\{f(A), c\} = \begin{cases} 0 & f(A) \geq c \\ c - f(A) & f(A) \leq c \leq f(A \cup \{x\}) \\ f(A \cup \{x\}) - f(A) & f(A) \leq f(A \cup \{x\}) \leq c \end{cases}$$

   Distinguiamo allora i tre casi:
   nel primo caso vale anche $f(B \cup \{x\}) \geq f(B) \geq c$ e quindi

   $$\min\{f(A \cup \{x\}), c\} - \min\{f(A), c\} = 0 = \min\{f(B \cup \{x\}), c\} - \min\{f(B), c\};$$

   nel secondo caso:

   $$f(A) \leq f(B) \implies c - f(A) \geq c - \min\{f(B), c\} = \min\{f(B \cup \{x\}), c\} - \min\{f(B), c\};$$

   nell'ultimo caso:

   $$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B) \geq \min\{f(B \cup \{x\}), c\} - \min\{f(B), c\}.$$

6. $\min\{f, g\}$ e $\max\{f, g\}$ non sono sempre submodulari. Infatti siano $f, g : \mathcal{P}(\{0, 1\}) \to \mathbb{R}$, e sia $f(S) = |S|$ e $g(S) = 2 - |S|$ $\forall S \in \mathcal{P}(\{0, 1\})$, allora detta $h = \max\{f, g\}$:

$$h(\emptyset) + h(\{0, 1\}) = 2 + 2 \not\leq 1 + 1 = h(\{0\}) + h(\{1\}).$$

1.2 Algoritmi di approssimazione e randomizzati

1.2.1 Algoritmi di approssimazione

Dato che per alcuni problemi non possiamo avere (o non conosciamo) un algoritmo che calcola la soluzione esatta in tempo polinomiale, è utile rilassare la richiesta: si richiede che l'algoritmo garantisca di dare una certa approssimazione del valore cercato. Nasce così la seguente definizione:

Definizione 1.3. Sia $\mathcal{A}$ un algoritmo per un problema di ottimizzazione P. Per semplicità si supponga che P consista nel massimizzare una funzione $f : X \to \mathbb{R}_{\geq 0}$ sull'insieme X delle possibili soluzioni.

Diciamo che $\mathcal{A}$ è **algoritmo di α-approssimazione** [12], con $\alpha \in (0, 1)$, se trova un valore $\tilde{x} \in X$ tale per cui $f(\tilde{x}) \geq \alpha OPT$ dove $OPT = \max_{x \in X} f(x)$ è il valore massimo di f.

Chiameremo inoltre α **fattore di approssimazione** dell'algoritmo $\mathcal{A}$ (rispetto al problema P).

Per i problemi di minimizzazione si ha una definizione analoga, con le uniche differenze che $\alpha \in (1, \infty)$ e che si richiede che la soluzione $\tilde{x} \in X$ sia tale che $f(\tilde{x}) \leq \alpha OPT$.

Si noti che la funzione f è supposta non negativa. Il motivo di questa precisazione sta nel fatto che se si avesse $OPT = \max_{x \in X} f(x) < 0$ allora $\forall \alpha \in (0, 1)$ e $\forall \tilde{x} \in X$ tale che $f(\tilde{x}) \geq \alpha OPT$ vorrebbe dire che $f(\tilde{x}) > OPT$. Il che è assurdo data la massimalità di OPT.

1.2.2 Algoritmi randomizzati

Spendiamo anche alcune parole sugli *algoritmi randomizzati*, in questi non tutte le istruzioni hanno un'esecuzione deterministica, ma dipendono anche da qualche evento aleatorio. È chiaro che questi sono una generalizzazione degli algoritmi classici, infatti ogni algoritmo deterministico è un algoritmo randomizzato in cui ogni evento è certo.

Anche un algoritmo randomizzato può essere di approssimazione per un certo problema di ottimizzazione P. Nel caso in cui questo consista nel massimizzare una funzione $f : X \to \mathbb{R}_{\geq 0}$, diremo che un algoritmo randomizzato $\mathcal{A}$ è **algoritmo randomizzato di α-approssimazione** se il valore atteso $\mathbb{E}[f(\tilde{x})] \geq \alpha OPT$, dove con $\tilde{x}$ indichiamo la soluzione che trova $\mathcal{A}$.

Capitolo 2

Algoritmi per la massimizzazione di funzioni submodulari

> An algorithm must be seen to be believed.
>
> D. E. Knuth

In tutto questo capitolo si considera il problema di massimizzare la funzione submodulare non negativa $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ su tutto il suo dominio. X è insieme finito di cardinalità n. Chiameremo inoltre $OPT = \max_{S \subseteq X} f(S)$.

Nella discussione di questi algoritmi supponiamo di trovarci nella seguente situazione modello, noto in letteratura come *value oracle model*. Si ha a disposizione un oracolo che permette di ottenere il valore della funzione f su un certo insieme $S \subseteq X$ in tempo costante. In questo modo nella discussione della complessità temporale degli algoritmi non inciderà il tempo necessario per la valutazione di f, che in certi casi potrebbe richiedere tempo anche esponenziale in n.

2.1 Un algoritmo non adattivo

Dimostriamo inizialmente un risultato per un algoritmo non adattivo, cioè in cui l'esecuzione dell'algoritmo non dipende dalla funzione in esame. Questo algoritmo è stato pubblicato e analizzato in [7].

L'algoritmo fissa un insieme $S \subseteq X$ scelto in modo uniformemente casuale su $\mathcal{P}(X)$, indipendentemente da f. Il risultato è semplicemente il valore $f(S)$.

Prima di dimostrare un teorema che ci mostra le garanzie date da questo algoritmo, dimostriamo il seguente lemma:

Lemma 2.1. *Sia $f : \mathcal{P}(X) \to \mathbb{R}$ funzione submodulare, e sia $p \in [0, 1]$, definisco l'insieme casuale $A(p) \subseteq A \subseteq X$ in cui $\mathbb{P}(x \in A(p)) = p \ \forall x \in A$ indipendentemente. Allora*

$$\mathbb{E}\left[f(A(p))\right] \geq (1-p)f(\emptyset) + pf(A).$$

Dimostrazione. Per induzione: il caso $|A| = 0$ è ovvio.

Si supponga il risultato vero per insiemi di cardinalità minore di k. Scriviamo $A = A' \cup \{x\}$ con $A' \subseteq \mathcal{P}(X)$ t.c. $|A'| = k-1$ e $x \in X \setminus A'$, e definiamo l'insieme casuale $A'(p) \subseteq A'$ come nell'enunciato, e $A(p) = A'(p)$ con probabilità $1-p$ e $A(p) = A'(p) \cup \{x\}$ altrimenti. Allora

$$\begin{aligned}\mathbb{E}\left[f(A(p))\right] &= \mathbb{E}\left[f(A(p)) - f(A'(p)) + f(A'(p))\right] \\ &\geq \mathbb{E}\left[f(A(p)) - f(A'(p))\right] + (1-p)f(\emptyset) + pf(A') \\ &\geq p(f(A) - f(A')) + (1-p)f(\emptyset) + pf(A') = (1-p)f(\emptyset) + pf(A).\end{aligned}$$

Per la prima disuguaglianza si è usata l'ipotesi induttiva assieme alla linearità del valore atteso; Per la seconda si è usato che: $\mathbb{E}\left[f(A(p)) - f(A'(p))\right] = p(f(A'(p) \cup \{x\}) - f(A'(p))) + (1-p)\cdot 0 \geq p(f(A) - f(A'))$, espandendo i due casi $x \in A(p)$ e $x \notin A(p)$. □

Possiamo quindi procedere con il teorema:

Teorema 2.2. *Sia $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ funzione submodulare. Dato un insieme $S \subseteq X$ casuale in cui $\mathbb{P}(x \in S) = \frac{1}{2}$ $\forall x \in X$ indipendentemente, allora*

$$\mathbb{E}\left[f(S)\right] = \frac{1}{4}OPT, \qquad OPT = \max_{R \subseteq X} f(R).$$

Inoltre se f è funzione submodulare simmetrica, si ha $\mathbb{E}\left[f(S)\right] = \frac{1}{2}f(OPT)$.

Dimostrazione. Sia $S \in \mathcal{P}(X)$ l'insieme scelto dall'algoritmo, e sia S^* un insieme t.c. $f(S^*) = OPT$. Allora scriviamo $S = (S \cap S^*) \cup (S \cap \overline{S^*})$, che, con la notazione del Lemma 2.1, posso scrivere come $S = S^*\left(\frac{1}{2}\right) \cup \overline{S^*}\left(\frac{1}{2}\right)$.

Ora condizioneremo i valori attesi rispetto ad un valore di $S^*\left(\frac{1}{2}\right) = S'$ fissato. Inoltre sappiamo che $Y \mapsto f(Y \cup S')$ è una funzione submodulare. Quindi usando il lemma troviamo che vale:

$$\mathbb{E}[f(S)] = \mathbb{E}\left[f\left(S' \cup \overline{S^*}\left(\frac{1}{2}\right)\right)\right] \geq \frac{1}{2}\mathbb{E}\left[f\left(S'\right)\right] + \frac{1}{2}\mathbb{E}\left[f\left(S' \cup \overline{S^*}\right)\right].$$

Ora rilassiamo il condizionamento su S', quindi utilizzando di nuovo il lemma sui due valori attesi troviamo:

$$\mathbb{E}[f(S)] \geq \frac{1}{4}f(\emptyset) + \frac{1}{4}f(S^*) + \frac{1}{4}f(\overline{S^*}) + \frac{1}{4}f(X) \geq \frac{1}{4}f(S^*) = \frac{1}{4}OPT.$$

Invece nel caso di funzione simmetrica la stima si può migliorare facilmente:

$$\mathbb{E}[f(S)] \geq \frac{1}{4}f(\emptyset) + \frac{1}{4}f(S^*) + \frac{1}{4}f(\overline{S^*}) + \frac{1}{4}f(X) = \frac{1}{2}f(S^*) + \frac{1}{2}f(X) \geq \frac{1}{2}OPT.$$

In entrambi i casi si è usato il fatto che f è una funzione non negativa. □

Il risultato appena dimostrato non può essere migliorato, nel senso che $\forall \varepsilon > 0\ \exists f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ t.c. il risultato dell'algoritmo sia $< \left(\frac{1}{4} + \varepsilon\right) OPT$. Questo si vedrà nel Capitolo 3.

È stupefacente che per le funzioni submodulari simmetriche questa approssimazione in tempo polinomiale è in realtà la migliore anche tra gli algoritmi adattivi, come mostrato all'inizio del Capitolo 3.

2.2 Un algoritmo greedy

In questa sezione sarà presentato un semplice algoritmo greedy di massimizzazione, come descritto in [7].

Sarà utile nella trattazione definire il concetto di ottimo locale approssimato:

Definizione 2.3. Sia $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ funzione submodulare, allora chiamiamo $S \subseteq X$ **massimo locale** $(1+\alpha)$**-approssimato** se vale:

$$f(S \cup \{x\}) \leq (1+\alpha) f(S) \quad \forall x \in X \setminus S \text{ e} \quad f(S \setminus \{x\}) \leq (1+\alpha) f(S) \quad \forall x \in S.$$

Proposizione 2.4. *Sia $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ funzione submodulare, e sia S massimo locale $(1+\alpha)$-approssimato. Allora $\forall I \subseteq X$ e $\forall J : S \subseteq J \subseteq X$ vale:*

$$f(I) \leq (1+n\alpha) f(S) \qquad f(J) \leq (1+n\alpha) f(S)$$

dove ricordiamo che $n = |X|$.

Dimostrazione. Mostriamo il risultato per $I \subseteq S$, per $J \supseteq S$ la dimostrazione si può fare in modo analogo. Definiamo alcuni insiemi $I =: I_0 \subseteq I_1 \subseteq \cdots \subseteq I_k := S$, in modo che $\forall i \in \{1, \ldots, k\}$ si abbia $I_i \setminus I_{i-1} = \{x_i\} \subset X$. Allora:

$$\forall i \in \{1, \ldots, k\} \qquad f(I_i) - f(I_{i-1}) \geq f(S) - f\left(S \setminus \{x_i\}\right) \geq -\alpha f(S),$$

per submodularità dato che $I_i \subseteq S$ per gli i considerati. Ora il risultato si trova con una somma telescopica:

$$f(S) - f(I) = \sum_{i=1}^{k} \left(f(I_i) - f(I_{i-1})\right) \geq -k\alpha f(S) \implies (1+k\alpha) f(S) \geq f(I).$$

Ricordando che $k \leq n$ allora $(1+n\alpha) f(S) \geq (1+k\alpha) f(S) \geq f(I)$. □

Si fissi $\varepsilon > 0$. Quindi definiamo l'algoritmo:

È chiaro che se questo algoritmo termina, allora al termine dell'esecuzione l'insieme S è massimo locale $\left(1 + \frac{\varepsilon}{n}\right)$-approssimato.

Teorema 2.5. *Se l'algoritmo 1 termina, esso trova un insieme S t.c.*

$$\max\{f(S), f(X \setminus S)\} \geq \left(\frac{1}{3} - \varepsilon\right) f(OPT).$$

Inoltre se f è submodulare simmetrica, allora la soluzione è una $\left(\frac{1}{2} - \varepsilon\right)$-approssimazione.

Algoritmo 1 Algoritmo greedy approssimato

1: $S \leftarrow \{v\}$ con v t.c. $f(\{v\})$ sia massimo tra i singoletti
2: **if** $\exists v : f(S \cup \{v\}) \geq \left(1 + \frac{\varepsilon}{n}\right) f(S)$ **then**
3: $\quad S \leftarrow S \cup \{v\}$ **go to** 2
4: **end if**
5: **if** $\exists v : f(S \setminus \{v\}) \geq \left(1 + \frac{\varepsilon}{n}\right) f(S)$ **then**
6: $\quad S \leftarrow S \setminus \{v\}$ **go to** 2
7: **end if**
8: **return** $\max\{f(S), f(X \setminus S)\}$

Dimostrazione. Sia $S^* \subseteq X$ t.c. $f(S^*) = OPT$.

L'insieme S trovato dall'algoritmo 1 al termine della sua esecuzione è un massimo locale $\left(1 + \frac{\varepsilon}{n}\right)$-approssimato. Allora usando prima la Proposizione 2.4 con $I = S \cap S^*$ e poi con $J = S \cup S^*$, e quindi sfruttando la submodularità due volte:

$$\begin{aligned} 2\left(1 + n\frac{\varepsilon}{n}\right) f(S) + f(X \setminus S) &\geq f(S \cap S^*) + f(S \cup S^*) + f(X \setminus S) \\ &\geq f(S \cap S^*) + f(X) + f(S^* \setminus S) \\ &\geq f(S \cap S^*) + f(S^* \setminus S) \geq f(S^*) + f(\emptyset). \end{aligned}$$

Ricordando che f è non negativa $f(\emptyset) \geq 0$, da qui si conclude che $2\left(1 + n\frac{\varepsilon}{n}\right) f(S) + f(X \setminus S) \geq f(S^*)$.

Ora consideriamo il valore ritornato dall'algoritmo 1: $V = \max\{f(S), f(X \setminus S)\}$. Per questa quantità vale:

$$(3 + 2\varepsilon)\, V \geq 2\,(1 + \varepsilon)\, f(S) + f(X \setminus S) \geq f(S^*) \implies V \geq \left(\frac{1}{3} - \varepsilon\right) OPT.$$

L'implicazione si giustifica spostando il fattore $(3 + 2\varepsilon)$ al membro di destra della disuguaglianza; a questo punto si ha la seconda disuguaglianza usando il fatto che $\frac{1}{3+2\varepsilon} \geq \frac{1}{3} - \frac{2\varepsilon}{9} \geq \frac{1}{3} - \varepsilon$, che vale perché $\varepsilon > 0$.

In particolare nel caso in cui f è funzione submodulare simmetrica, usando la Proposizione 2.4 e poi la submodularità in modo analogo a sopra, si vede che:

$$(2 + \varepsilon)\, V \geq (1 + \varepsilon)\, f(S) + f(X \setminus S) \geq f(S \cup S^*) + f(X \setminus S) \geq f(X \setminus S^*) = f(S^*).$$

Questo implica, in modo analogo al caso generale, che $f(S) \geq \left(\frac{1}{2} - \varepsilon\right) OPT$. □

Fino ad ora non abbiamo avuto la certezza che l'algoritmo 1 termini. Non è però difficile vedere che esso deve terminare in tempo polinomiale, come mostrato nella seguente proposizione.

Proposizione 2.6. *Il numero di valutazioni della funzione f che fa l'algoritmo 1 è $\mathcal{O}(\frac{1}{\varepsilon} n^2 \log n)$.*

Dimostrazione. Si consideri $x^* \in X$ t.c. $f(\{x^*\}) = \max_{x \in X} f(\{x\})$. Allora per submodularità $OPT \leq nf(\{x^*\})$.

Nell'algoritmo per ogni variazione di S, il valore $f(S)$ aumenta di un fattore di almeno $\left(1 + \frac{\varepsilon}{n}\right)$, e quindi in k iterazioni aumenta di un fattore $\geq \left(1 + \frac{\varepsilon}{n}\right)^k$. Quindi per il valore di S trovato alla k-esima iterazione vale:

$$\left(1 + \frac{\varepsilon}{n}\right)^k f(\{x^*\}) \leq f(S) \leq OPT \leq nf(\{x^*\}) \implies k = \mathcal{O}\left(\frac{1}{\varepsilon} n \log n\right).$$

Per ogni iterazione, f viene valutata in al più $2n+1$ insiemi distinti, quindi il numero totale di valutazioni di f è al più $(2n+1)\mathcal{O}\left(\frac{1}{\varepsilon} n \log n\right) = \mathcal{O}\left(\frac{1}{\varepsilon} n^2 \log n\right)$. □

2.3 Due algoritmi lineari

In questa sezione saranno descritti due algoritmi lineari approssimati per il massimo di una funzione submodulare. I due algoritmi sono stati pubblicati in [3]. Il secondo di questi è una versione randomizzata del primo e raggiunge il limite di approssimazione per gli algoritmi polinomiali.

Anche questi algoritmi utilizzano una strategia greedy; l'idea nuova è che vengono ottimizzati i valori della funzione su due insiemi contemporaneamente, uno "piccolo" e uno "grande"; questi vengono progressivamente modificati un elemento alla volta fino a renderli uguali.

Per questa sezione è utile dare un nome a tutti gli elementi di X, quindi diciamo che $X = \{x_1, \ldots, x_n\}$. Si noti che non è importante l'ordine di questi x_i per $i = 1, \ldots, n$.

2.3.1 Un algoritmo lineare deterministico

In questa prima parte della sezione sarà descritto un algoritmo di approssimazione deterministico che impiega un tempo lineare nella cardinalità di X. Nonostante il suo fattore di approssimazione sia soltanto $\frac{1}{3}$, conoscere questo algoritmo semplifica la comprensione dell'algoritmo 3, che è una versione randomizzata dello stesso algoritmo.

Prima di procedere è utile osservare che gli insiemi A_k e B_k, come calcolati dall'algoritmo 2, coincidono su $\{x_1, \ldots, x_k\}$, $\forall k \in \{1, \ldots, n\}$. Infatti prima della i-esima iterazione $x_i \notin A_{i-1}$ e $x_i \in B_{i-1}$, ma durante l'iterazione x_i viene inserito in A_i oppure x_i è tolto da B_i, e non accadono mai entrambe.

Lemma 2.7. *Per i valori a_i e b_i come definiti nell'algoritmo 2 vale $a_i + b_i \geq 0$ $\forall i \in \{1, \ldots, n\}$.*

Algoritmo 2 Algoritmo lineare di $\frac{1}{3}$-approssimazione

$A_0 \leftarrow \emptyset$
$B_0 \leftarrow X$
for $i \in \{1, \dots, n\}$ **do**
    $a_i \leftarrow f(A_{i-1} \cup \{x_i\}) - f(A_{i-1})$
    $b_i \leftarrow f(B_{i-1} \setminus \{x_i\}) - f(B_{i-1})$
    if $a_i \geq b_i$ **then** $A_i \leftarrow A_{i-1} \cup \{x_i\}$; $B_i \leftarrow B_{i-1}$
    else $B_i \leftarrow B_{i-1} \setminus \{x_i\}$; $A_i \leftarrow A_{i-1}$
    end if
end for
return $f(A_n)$

Dimostrazione. Usando la submodularità di f si ha:

$$\begin{aligned} a_i + b_i &= f(A_{i-1} \cup \{x_i\}) - f(A_{i-1}) + f(B_{i-1} \setminus \{x_i\}) - f(B_{i-1}) \\ &= [f(A_{i-1} \cup \{x_i\}) + f(B_{i-1} \setminus \{x_i\})] - f(A_{i-1}) - f(B_{i-1}) \\ &\geq f(A_{i-1}) + f(B_{i-1}) - f(A_{i-1}) - f(B_{i-1}) = 0. \end{aligned}$$

Infatti $A_{i-1} = (A_{i-1} \cup \{x_i\}) \cap (B_{i-1} \setminus \{x_i\})$ e $B_{i-1} = (A_{i-1} \cup \{x_i\}) \cup (B_{i-1} \setminus \{x_i\})$, perché A_{i-1} e B_{i-1} contengono gli stessi x_j per $j \leq i-1$, e rispettivamente contengono nessuno e tutti gli x_j per $j > i-1$. □

Per il prossimo lemma sarà necessario definire alcuni insiemi $S_0^*, \dots, S_n^*$. Supponiamo di avere un insieme S^* t.c. $f(S^*) = OPT$ cioè il massimo di f, allora definiamo $S_i^* := (S^* \cup A_i) \cap B_i$. Quindi ogni insieme S_i^* coincide con A_i e B_i sui primi i elementi di X, e con S^* sui restanti elementi di X.

Lemma 2.8. *Per gli insiemi A_i, B_i, S_i^* definiti sopra e calcolati dall'algoritmo 2, vale:*

$$f(S_{i-1}^*) - f(S_i^*) \leq f(A_i) - f(A_{i-1}) + f(B_i) - f(B_{i-1}) \qquad \forall i \in \{1, \dots, n\}.$$

Dimostrazione. Sia $i \in \{1, \dots, n\}$ fissato. Supponiamo che valga $a_i \geq b_i$, il caso opposto è analogo. In questo caso abbiamo allora $B_i = B_{i-1}$ e $A_i = A_{i-1} \cup \{x_i\}$. Quindi va dimostrato che:

$$f(S_{i-1}^*) - f(S_{i-1}^* \cup \{x_i\}) \leq f(A_i) - f(A_{i-1}) = a_i.$$

Distinguiamo due casi:

$$x_i \in S^* \implies f(S_{i-1}^*) - f(S_{i-1}^* \cup \{x_i\}) = 0 \leq f(A_i) - f(A_{i-1}) = a_i$$

dato che $a_i + b_i \geq 0$ per il primo lemma e $a_i \geq b_i$. Mentre l'altro caso:

$$x_i \notin S^* \implies f(S_{i-1}^*) - f(S_{i-1}^* \cup \{x_i\}) \leq f(B_{i-1} \setminus \{x_i\}) - f(B_{i-1}) = b_i \leq a_i$$

Che è vero per submodularità, dato che $S_{i-1} \subseteq B_{i-1} \setminus \{x_i\}$. □

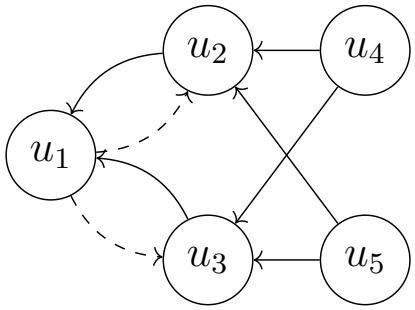

Figura 2.1: In questo grafo gli archi pieni hanno peso 1, mentre quelli tratteggiati $1 - \varepsilon$. Se si vuole trovare il massimo taglio, usando l'algoritmo 2 trova sempre un'approssimazione $\leq \frac{1}{3} + \varepsilon$ del massimo globale.

Teorema 2.9. *L'algoritmo 2 produce una $\frac{1}{3}$-approssimazione del massimo globale, cioè:*

$$f(A_n) \geq \frac{1}{3} OPT$$

Dimostrazione. Usiamo il lemma precedente sulla seguente somma telescopica:

$$\begin{aligned} f(S^*) - f(S_n^*) = \sum_{i=1}^{n} f(S_{i-1}^*) - f(S_i^*) &\leq \sum_{i=1}^{n} [f(A_i) - f(A_{i-1}) + f(B_i) - f(B_{i-1})] \\ &= f(A_n) - f(A_0) + f(B_n) - f(B_0) \leq 2f(A_n). \end{aligned}$$

L'ultima disuguaglianza è vera dato che f è non negativa, e ricordando che $A_n = B_n = S_n^*$. Quindi $OPT = f(S^*) \leq 3f(A_n) \implies \frac{1}{3} OPT \leq f(A_n)$. □

Ora che abbiamo trovato una garanzia sul risultato dell'algoritmo 2, è interessante osservare che il fattore di approssimazione trovato è realmente il massimo che quest'algoritmo garantisce. Per provare questo è sufficiente osservare il seguente esempio.

Esempio 1. Si fissi un $\varepsilon \in (0, 1)$. Si consideri la funzione di taglio $f : X = \{u_1, u_2, u_3, u_4, u_5\} \to \mathbb{R}$ sul grafo descritto in figura 1, con i pesi sugli archi descritti nella didascalia.

Seguiamo quindi l'esecuzione dell'algoritmo: inizialmente abbiamo $A_0 = \emptyset$, e $B_0 = \{u_1, u_2, u_3, u_4, u_5\}$, ed osserviamo singolarmente ogni iterazione, considerando i nodi dell'ordine u_1, u_2, u_3, u_4, u_5:

1. abbiamo $a_1 = 2 - 2\varepsilon$ e $b_1 = 2$, quindi $A_1 = \emptyset$, $B_1 = \{u_2, u_3, u_4, u_5\}$;
2. abbiamo $a_2 = 1 = b_2$, quindi $A_2 = \{u_2\}$, $B_2 = \{u_2, u_3, u_4, u_5\}$;
3. analogamente al precedente, otteniamo: $A_3 = \{u_2, u_3\}$, $B_3 = \{u_2, u_3, u_4, u_5\}$;
4. aggiungere u_4 a A_3 non fa guadagnare nulla, ma nemmeno togliere u_4 da B_3, quindi abbiamo: $A_4 = \{u_2, u_3, u_4\}$, $B_4 = \{u_2, u_3, u_4, u_5\}$;

5. analogamente al precedente, si ottiene: $A_5 = B_5 = \{u_2, u_3, u_4, u_5\}$;

Quindi l'algoritmo 2 calcola il massimo approssimato $f(\{u_1, u_2, u_3, u_4, u_5\}) = 2$.

Si osservi però che il massimo globale è $OPT = 6 - 2\varepsilon = f(\{u_1, u_4, u_5\})$. Questo vuol dire che l'approssimazione del massimo trovata dall'algoritmo è $\frac{2}{6-2\varepsilon} OPT \leq \left(\frac{1}{3} + \varepsilon\right) OPT$. Questo mostra che le garanzie sul risultato di questo algoritmo viste nei precedenti teoremi non sono migliorabili.

2.3.2 Un algoritmo lineare randomizzato

L'idea dell'algoritmo 3, presentato in questa sottosezione, è di ponderare in modo più equilibrato rispetto all'algoritmo 2 le due possibilità ad ogni iterazione. Questo viene fatto in modo abbastanza naturale introducendo aleatorietà nell'esecuzione. In questo modo si riesce ad ottenere in valore atteso una $\frac{1}{2}$-approssimazione dell'ottimo.

Algoritmo 3 Algoritmo lineare di $\frac{1}{2}$-approssimazione

1: $A_0 \leftarrow \emptyset$
2: $B_0 \leftarrow X$
3: **for** $i \in \{1, \ldots, n\}$ **do**
4: $\quad a_i \leftarrow \max\{0, f(A_{i-1} \cup \{x_i\}) - f(A_{i-1})\}$
5: $\quad b_i \leftarrow \max\{0, f(B_{i-1} \setminus \{x_i\}) - f(B_{i-1})\}$
6: $\quad$ Se $a_i + b_i = 0$ oppure con probabilità $\frac{a_i}{a_i+b_i}$: $A_i \leftarrow A_{i-1} \cup \{x_i\}$; $B_i \leftarrow B_{i-1}$
7: $\quad$ Altrimenti $\left(\text{con probabilità } \frac{b_i}{a_i+b_i}\right)$ $B_i \leftarrow B_{i-1} \setminus \{x_i\}$; $A_i \leftarrow A_{i-1}$
8: **end for**
9: **return** $f(A_n)$

Definiamo, come per l'algoritmo 2, gli insiemi $S_i^* := (S^* \cup A_i) \cap B_i$ per $i = 0, \ldots, n$, con S^* come al solito t.c. $f(S^*) = OPT$.

Lemma 2.10. *Per gli insiemi A_i, B_i, S_i^*, per $i = 1, \ldots, n$, definiti sopra e calcolati dall'algoritmo 3, vale:*

$$\mathbb{E}[f(S_{i-1}^*) - f(S_i^*)] \leq \frac{1}{2}\mathbb{E}[f(A_i) - f(A_{i-1}) + f(B_i) - f(B_{i-1})].$$

Dimostrazione. Fissiamo un A_{i-1}, e per tutta la dimostrazione condizioniamo le probabilità rispetto a questo evento.

Distinguiamo alcuni casi:

Caso $a_i \geq 0, b_i = 0$: allora $A_i = A_{i-1} \cup \{x_i\}$, $B_i = B_{i-1}$. Quindi se $x_i \in S^*$ abbiamo che il membro di sinistra si azzera, quindi la relazione è banalmente soddisfatta. Altrimenti se $x_i \notin S^*$ vale:

$$f(S_{i-1}^*) - f(S_{i-1}^* \cup \{x_i\}) \leq f(B_{i-1} \setminus \{x_i\}) - f(B_{i-1}) \leq b_i = 0 \leq \frac{f(A_i) - f(A_{i-1})}{2} = \frac{a_i}{2}$$

per submodularità.

Caso $a_i = 0, b_i \geq 0$ si procede analogamente al caso precedente: abbiamo $A_i = A_{i-1}$, $B_i = B_{i-1} \setminus \{x_i\}$. Se $x_i \notin S^*$ allora il membro di sinistra si azzera. Altrimenti $x_i \in S^*$ e per submodularità vale:

$$f(S^*_{i-1}) - f(S^*_{i-1} \setminus \{x_i\}) \leq f(A_{i-1} \cup \{x_i\}) - f(A_{i-1}) \leq a_i = 0 \leq \frac{f(B_i) - f(B_{i-1})}{2} = \frac{b_i}{2}$$

Caso $a_i, b_i > 0$: Per prima cosa espandiamo il valore atteso del membro di destra della tesi:

$$\begin{aligned}\mathbb{E}[f(A_i) - f(A_{i-1}) + f(B_i) - f(B_{i-1})] = \frac{a_i}{a_i + b_i}\left[f(A_{i-1} \cup \{x_i\}) - f(A_{i-1})\right] + \\ + \frac{b_i}{a_i + b_i}\left[f(B_{i-1} \setminus \{x_i\}) - f(B_{i-1})\right] = \frac{a_i^2 + b_i^2}{a_i + b_i}.\end{aligned}$$

Mentre al membro di sinistra:

$$\begin{aligned}\mathbb{E}[f(S^*_{i-1}) - f(S^*_i)] = \frac{a_i}{a_i + b_i}\left[f(S^*_{i-1}) - f(S^*_{i-1} \cup \{x_i\})\right] + \\ + \frac{b_i}{a_i + b_i}\left[f(S^*_{i-1}) - f(S^*_{i-1} \setminus \{x_i\})\right] \qquad \leq \frac{a_i b_i}{a_i + b_i}.\end{aligned}$$

L'ultima disuguaglianza è vera perché uno dei due termini è nullo, mentre per l'altro si fa la stima analogamente ai casi precedenti.

A questo punto si può concludere usando la disuguaglianza tra media geometrica e quadratica:

$$\mathbb{E}[f(S^*_{i-1}) - f(S^*_i)] \leq \frac{a_i b_i}{a_i + b_i} \leq \frac{1}{2}\frac{a_i^2 + b_i^2}{a_i + b_i} = \frac{1}{2}\mathbb{E}[f(A_i) - f(A_{i-1}) + f(B_i) - f(B_{i-1})].$$

Con questo si sono esauriti i casi possibili. □

Proseguiamo quindi con il risultato principale:

Teorema 2.11. *L'algoritmo 3 ha risultato in valore atteso* $\mathbb{E}\left[f(A_n)\right] \geq \frac{1}{2}OPT$.

Dimostrazione. Procediamo in modo analogo al corrispondente teorema per l'algoritmo 2. È sufficiente osservare alcune somme telescopiche:

$$\begin{aligned}\mathbb{E}[f(S^*) - f(S^*_n)] &= \sum_{i=1}^{n} \mathbb{E}[f(S^*_{i-1}) - f(S^*_i)] \\ &\leq \sum_{i=1}^{n} \frac{1}{2}\mathbb{E}\left[f(A_i) - f(A_{i-1}) + f(B_i) - f(B_{i-1})\right] \\ &\leq \frac{1}{2}\mathbb{E}\left[f(A_n) - f(A_0) + f(B_n) - f(B_0)\right].\end{aligned}$$

Quindi usando la linearità del valore atteso, la non negatività di f e ricordando che $A_n = B_n = S^*_n$, si vede che $\frac{1}{2}f(S^*) \leq \mathbb{E}\left[f(A_n)\right]$. □

Abbiamo così confermato che l'algoritmo 3 è un algoritmo randomizzato di $\frac{1}{2}$-approssimazione per la massimizzazione di funzioni submodulari senza vincoli. Nel prossimo capitolo si confermerà che questo algoritmo è ottimo, nel senso che non può esistere un algoritmo che garantisce una approssimazione migliore per tutte le funzioni submodulari in tempo polinomiale.

Capitolo 3

Limiti della massimizzazione

> It's hard to drive at the limit, but it's harder to know where the limits are.
>
> Stirling Moss

I limiti che dimostreremo riguardano il numero di valutazioni della funzione che si vuole massimizzare. In particolare se $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ è la funzione in questione, con $|X| = n \in \mathbb{N}$, allora un algoritmo che garantisce un fattore di approssimazione strettamente migliore di un certo valore, deve valutare f in almeno un numero di insiemi esponenziale all'aumentare di n. Tali risultati hanno una chiara implicazione di complessità nel *value oracle model*, ma non necessariamente in altri modelli.

Dimostriamo inizialmente un risultato che prova l'ottimalità del risultato dato per il modello *random set*, descritto nella sezione 2.1, così come fatto in [7].

Teorema 3.1. *Sia X insieme con $|X| = n \in \mathbb{N}$. Allora se n è sufficientemente grande, $\forall \varepsilon > 0, \exists \delta > 0$ tale che per ogni insieme $\mathcal{Q} \subseteq \mathcal{P}(X)$, $|\mathcal{Q}| \leq 2^{\delta n}$ esiste una funzione submodulare $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ t.c.*

$$f(Q) \leq \left(\frac{1}{4} + \varepsilon\right) OPT \quad \forall Q \in \mathcal{Q} \qquad OPT = \max_{S \subseteq X} f(S).$$

Dimostrazione. Osserviamo preliminarmente che si può supporre senza perdita di generalità che $\varepsilon > 0$ sia piccolo a piacere, dato che se il teorema è vero con $\varepsilon = \varepsilon_0 > 0$ allora è vero per tutti gli $\varepsilon > \varepsilon_0$.

Usiamo il metodo probabilistico: quello che faremo è dimostrare che, fissato ε abbiamo un $\delta > 0$ t.c. per ogni $\mathcal{Q}$ come nelle ipotesi, se consideriamo una classe di funzioni, la probabilità che una funzione presa casualmente da questo insieme abbia fattore di approssimazione $> (\frac{1}{4} + \varepsilon)$ è strettamente minore di 1.

Prendiamo f_C come funzione di taglio su un grafo diretto bipartito completo di parti $C \subseteq X, \overline{C} = X \setminus C$. Prendiamo C uniformemente aleatorio in $\mathcal{P}(X)$. f_C è allora $f_C(A) = |A \cap C| \cdot |A \setminus C|$, e ha massimo $OPT = f_C(C) = |C| \cdot |\overline{C}|$.

Consideriamo quindi un insieme $A \subseteq X$ qualunque. Allora per i *Chernoff bound* (descritti in Appendice A) si hanno le due disuguaglianze:

$$\mathbb{P}\left(|A \cap C| \geq (1+\varepsilon)\frac{|A|}{2}\right) \leq e^{-\frac{\varepsilon^2|A|}{3}} \qquad \mathbb{P}\left(|A \cap C| \leq (1-\varepsilon)\frac{|A|}{2}\right) \leq e^{-\frac{\varepsilon^2|A|}{3}}. \quad (3.1)$$

Allora se prendiamo $A = X$ abbiamo che con probabilità almeno $1 - 2e^{-\frac{\varepsilon^2 n}{3}}$ è vero che $|A \cap C| = |X \cap C| = |C| \in \left[(1-\varepsilon)\frac{n}{2}, (1+\varepsilon)\frac{n}{2}\right]$, quindi è lecito restringere la scelta di C agli insiemi tali che $|C| \in \left[(1-\varepsilon)\frac{n}{2}, (1+\varepsilon)\frac{n}{2}\right]$. Dunque per ε abbastanza piccolo vale $OPT \geq \frac{(1-\varepsilon)(1+\varepsilon)}{4}n^2 \geq \frac{n^2}{4(1+\varepsilon)}$.

Ora si vede che $f_C(Q) > \left(\frac{1}{4}+\varepsilon\right)OPT$ implica $f_C(Q) > \frac{n^2}{16}$, quest'ultima disuguaglianza può essere vera soltanto nel caso in cui $|Q| \in \left[\frac{n}{16}, \frac{15n}{16}\right]$. Supponiamo allora che questo sia vero $\forall Q \in \mathcal{Q}$. Da questo deduco, usando la stima 3.1, che:

$$\mathbb{P}\left(|Q \cap C| \geq (1+\varepsilon)\frac{|Q|}{2}\right) \leq e^{-\varepsilon^2|Q|/3} \leq e^{-\varepsilon^2 n/48}$$

e analogamente $\mathbb{P}\left(|Q \setminus C| \geq (1+\varepsilon)\frac{|\overline{Q}|}{2}\right) = \mathbb{P}\left(|\overline{C} \cap \overline{Q}| \geq (1+\varepsilon)\frac{|\overline{Q}|}{2}\right) \leq e^{-\varepsilon^2 n/48}$.

Ora nel caso in cui $|Q \cap C| < (1+\varepsilon)\frac{|Q|}{2}$ e $|Q \setminus C| < (1+\varepsilon)\frac{|\overline{Q}|}{2}$, evento che avviene con probabilità almeno $1 - 2e^{\varepsilon^2 n/48}$, ho:

$$f_C(Q) = |C \cap Q| \cdot |Q \setminus C| < \frac{(1+\varepsilon)^2}{4}|Q||\overline{Q}| \leq \frac{n^2}{16}.$$

Allora $\mathbb{P}\left(f_C(Q) \leq \frac{n^2}{16}\right) \geq \mathbb{P}\left(|Q \cap C| < (1+\varepsilon)\frac{|Q|}{2}, |Q \setminus C| < (1+\varepsilon)\frac{|\overline{Q}|}{2}\right) \geq 1 - 2e^{\varepsilon^2 n/48}$, di conseguenza $\mathbb{P}\left(f_C(Q) > \frac{n^2}{16}\right) \leq 2e^{\varepsilon^2 n/48}$.

Quindi la probabilità che almeno uno dei $Q \in \mathcal{Q}$ sia t.c. $f_C(Q) > \left(\frac{1}{4}+\varepsilon\right)OPT$ è stimata dall'alto da $|\mathcal{Q}| \cdot 2e^{-\varepsilon^2 n/48} \leq 2^{\delta n} \cdot 2e^{-\varepsilon^2 n/48}$, dato che $\mathbb{P}\left(f_C(Q) > (\frac{1}{4}+\varepsilon)OPT\right) \leq \mathbb{P}\left(f_C(Q) > \frac{n^2}{16}\right) \leq 2e^{\varepsilon^2 n/48}$. Possiamo ora fissare $\delta = \frac{\varepsilon^2}{48}$. Allora, stimando la probabilità dell'unione di eventi come la somma delle probabilità di ciascun evento, otteniamo che per n sufficientemente grande:

$$\mathbb{P}\left(\max_{Q \in \mathcal{Q}} f_C(Q) > \left(\frac{1}{4}+\varepsilon\right)OPT\right) \leq |\mathcal{Q}| \cdot 2e^{\varepsilon^2 n/48} \leq 2\left(\frac{2}{e}\right)^{\delta n} < 1.$$

Quindi abbiamo dimostrato la tesi, cioè che $\exists C \subseteq X$ e quindi una funzione f_C tale per cui gli insiemi $\mathcal{Q}$ non raggiungono il limite. □

Teorema 3.2. *Per ogni $\varepsilon > 0$ e per ogni algoritmo (anche randomizzato) che massimizza funzioni submodulari con meno di $e^{\varepsilon^2 n/16}$ valutazioni, esiste una funzione submodulare $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ (con $|X| = n \in \mathbb{N}$) tale per cui la soluzione calcolata ha valore $< \left(\frac{1}{2}+\varepsilon\right)OPT$.*

Dimostrazione. Si consideri una funzione submodulare $f : \mathcal{P}(X) \to \mathbb{R}$ tale che:

- $f(S)$ dipenda solo da due valori $k = |S \cap C|$ e $\ell = |S \cap D|$, con C e D disgiunti e fissati t.c. $|C| = |D| = \frac{n}{2}$. Allora ha senso scrivere $f(S) = f(k, \ell)$.
- Per $|k - \ell| \leq \varepsilon n$ si abbia $f(k, \ell) = (k + \ell)(n - k - \ell)$, cioè f ha il valore della funzione di taglio sul un grafo completo. In questo caso il massimo è $\frac{1}{4}n^2$.
- Per $|k - \ell| > \varepsilon n$ si abbia $f(k, \ell) = k(n - 2\ell) + \ell(n - 2k) - \mathcal{O}(\varepsilon n^2)$, cioè f ha valore vicino alla funzione di taglio sul un grafo bipartito completo con peso 2 su ogni arco. In questo caso il massimo è nell'ordine di $\frac{1}{2}n^2(1 - \mathcal{O}(\varepsilon))$, che viene raggiunto per $k = n/2$, $\ell = 0$ (o viceversa).

Allora una tale funzione è:

$$f(k, \ell) := \begin{cases} (k + \ell)(n - k - \ell) = |S|\,(n - |S|) & \text{se } |k - \ell| \leq \varepsilon n; \\ k(n - 2\ell) + \ell(n - 2k) - 2\varepsilon n|k - \ell| + \varepsilon^2 n^2 & \text{se } |k - \ell| > \varepsilon n. \end{cases}$$

Si osservi che l'aggiunta del termine $\varepsilon^2 n^2$ garantisce che la f sia "liscia", cioè per $k - \ell = \pm\varepsilon n$ le due espressioni combaciano: $(k+\ell)(n-k-\ell) = (2k \mp \varepsilon n)(n - 2k \pm \varepsilon n) = k(n - 4k \pm \varepsilon n) + (k \mp \varepsilon n)(n - 2k) - \varepsilon^2 n^2 = k(n - 2\ell) + \ell(n - 2k) - \varepsilon^2 n^2$.

L'unica cosa da verificare è che questa funzione sia submodulare. Per vedere questo è sufficiente calcolare le derivate direzionali nei diversi casi:

$$f(k + 1, \ell) - f(k, \ell) = \begin{cases} n - 4\ell + 2\varepsilon n & \text{se } k - \ell < -\varepsilon n; \\ n - 2k - 2\ell - 1 & \text{se } |k - \ell| \leq \varepsilon n; \\ n - 4\ell - 2\varepsilon n & \text{se } k - \ell > \varepsilon n. \end{cases}$$

Infatti per il primo caso si trova che la derivata diventa:

$$(n - 2\ell) - 2\ell + 2\varepsilon n = (1 + 2\varepsilon)n - 4\ell.$$

Similmente vale il terzo caso. Mentre per il secondo caso vale:

$$(k+1+\ell)(n-k-1-\ell) - (k+\ell)(n-k-\ell) = (n-k-1-\ell) - (k+\ell) = n - 2k - 2\ell - 1.$$

Queste derivate sono decrescenti all'aumentare di k ed ℓ.

Le derivate direzionali in ℓ sono analoghe, dato che $f(k, \ell) = f(\ell, k)$.

Nota questa costruzione, il resto della dimostrazione segue un'idea simile a quanto fatto nel Teorema 3.1. Per gli stessi motivi esposti nella dimostrazione di tale teorema, è lecito supporre che $\varepsilon > 0$ sia piccolo a piacere.

Si prenda un algoritmo qualsiasi come nell'enunciato, limitiamoci per ora al caso in cui l'algoritmo è deterministico.

Chiamiamo un insieme Q "bilanciato" se $||C \cap Q| - |D \cap Q|| = |k - \ell| \leq \varepsilon n$, e prendiamo C aleatorio uniforme su $\mathcal{P}(X)$. Chiamiamo anche E l'evento "tutti gli insiemi in cui è valutata la funzione sono bilanciati".

Usando il *Chernoff Bound* come nell'equazione 3.1 abbiamo che la probabilità che un certo insieme Q non sia bilanciato è almeno $e^{-\varepsilon^2 n/8}$. Quindi se per ipotesi ci sono al più $e^{\varepsilon^2 n/16}$ insiemi su cui valuto la funzione, vale $\mathbb{P}(E) \geq 1 - e^{\varepsilon^2 n/16} \cdot e^{-\varepsilon^2 n/8} = 1 - e^{-\varepsilon^2 n/16} > 0$.

Nel caso di un algoritmo randomizzato, condizioniamo l'esecuzione dell'algoritmo rispetto allo stato della fonte di aleatorietà. In questo modo l'algoritmo si può trattare come nel caso deterministico, con risultato analogo a sopra: $\mathbb{P}(E) \geq 1 - e^{-\varepsilon^2 n/16} > 0$. Ma questo vuol dire che rilassando il condizionamento vale anche la disuguaglianza $\mathbb{E}[\mathbb{P}(E)] \geq 1 - e^{-\varepsilon^2 n/16} > 0$, dato che $\mathbb{P}(E) \geq 1 - e^{-\varepsilon^2 n/16}$ per ogni stato della fonte di aleatorietà.

Ora se tutti gli insiemi valutati sono bilanciati, l'esecuzione dell'algoritmo è la stessa per f e per $g(S) = |S|(n - |S|)\ \forall S \subseteq X$, e quindi il risultato è lo stesso. Però g ha massimo che vale $\frac{1}{4}n^2$, mentre f ha massimo $\frac{1}{2}(1 - \mathcal{O}(\varepsilon))n^2$.

Quindi il fattore di approssimazione che si ottiene nell'esecuzione dell'algoritmo è $\frac{\frac{1}{4}n^2}{\frac{1}{2}(1-\mathcal{O}(\varepsilon))n^2} = \frac{1}{2}\frac{1}{1-\mathcal{O}(\varepsilon)} < \left(\frac{1}{2} + \varepsilon\right)$. □

È interessante anche sapere che si può dimostrare che se esistesse un algoritmo polinomiale che massimizza esattamente una funzione submodulare allora sarebbe $P = NP$ [7].

Capitolo 4

Algoritmi per la massimizzazione con vincolo di cardinalità

In questo capitolo tratteremo un problema più generale, ovvero quello di massimizzare una funzione submodulare $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ sugli insiemi di cardinalità al più $k \in \mathbb{N}$, cioè su un insieme $C = \{S \in \mathcal{P}(X) : |S| \leq k\}$ invece che tutto $\mathcal{P}(X)$. Per semplicità supponiamo che sia $k \leq n = |X|$.

In modo analogo ai capitoli precedenti chiamiamo $OPT = \max_{S \in C} f(S)$ il valore ottimo, e chiamiamo $S^* \subseteq X$ t.c. $f(S^*) = OPT$ e $|S^*| \leq k$.

4.1 Un algoritmo per funzioni submodulari monotone

Cominciamo trattando il caso più semplice in cui f sia monotona. In questo caso un semplice algoritmo greedy già garantisce un fattore di approssimazione pari a $1 - \frac{1}{e}$. Questo valore è il migliore che si può raggiungere in tempo polinomiale, infatti per il problema più specifico di *max-k-cover* questo è il miglior fattore di approssimazione ottenibile in tempo polinomiale, a meno che $P = NP$ [6]. Tale dimostrazione non verrà riproposta in questa tesi.

Algoritmo 4 Algoritmo greedy per funzioni monotone con vincolo di cardinalità

1: $S_0 \leftarrow \emptyset$
2: **for** $i \in \{1, \ldots, k\}$ **do**
3: $\quad S_i \leftarrow S_{i-1} \cup \{\arg\max_{v \in X \setminus S_{i-1}} [f(S_{i-1} \cup \{v\}) - f(S_{i-1})]\}$
4: **end for**
5: **return** $f(S_k)$

Riportiamo allora un semplice risultato, che quantifica il fattore di approssimazione che quest'algoritmo offre. La dimostrazione è tratta da [10], ma il teorema è stato dimostrato per la prima volta in [11].

Teorema 4.1. *L'algoritmo 4 produce come risultato un valore $f(S_k)$ t.c.*

$$f(S_k) \geq \left(1 - \frac{1}{e}\right) OPT.$$

Dimostrazione. Scegliamo S^* tale che $f(S^*) = OPT$, allora possiamo supporre anche che $|S^*| = k$ dato che f è monotona. Chiamiamo quindi $S^* = \{s_1, \ldots, s_k\} \in \mathcal{P}(X)$ un tale insieme. Ora $\forall i \in \{0, \ldots, k-1\}$ abbiamo le seguenti disuguaglianze:

$$f(S^*) \leq f(S_i \cup S^*) = f(S_i) + \sum_{i=1}^{k} \left(f(S_i \cup \{s_1, \ldots, s_i\}) - f(S_i \cup \{s_1, \ldots, s_{i-1}\})\right)$$

$$\leq f(S_i) + \sum_{i=1}^{k} \left(f(S_i \cup \{s_i\}) - f(S_i)\right) \leq f(S_i) + k\left(f(S_{i+1}) - f(S_i)\right).$$

Queste disuguaglianze sono giustificate, nell'ordine, dalla monotonia di f; dalla submodularità; e dalla massimalità di $f(S_i) - f(S_{i-1})$ dovuta alla scelta dell'algoritmo. L'uguaglianza è una semplice somma telescopica.

Definiamo allora $\delta_i := f(S^*) - f(S_i)$. Così la disuguaglianza $f(S^*) - f(S_i) \leq k\left(f(S_{i+1}) - f(S_i)\right)$ si può riscrivere in modo equivalente come:

$$\delta_i \leq k(\delta_i - \delta_{i+1}) \iff \delta_{i+1} \leq \left(1 - \frac{1}{k}\right) \delta_i.$$

Da questo si vede che $\delta_{i+1} \leq \left(1 - \frac{1}{k}\right)^{i+1} \delta_0$. E ricordo che $\delta_0 = f(S^*) - f(\emptyset) \leq f(S^*) = OPT$ dato che f è non negativa.

Usiamo allora la nota disuguaglianza $1 + x \leq e^x$, che vale $\forall x \in \mathbb{R}$. Per $x = -\frac{1}{k}$ abbiamo $1 - \frac{1}{k} \leq e^{-1/k}$. Con questo fatto si vede che:

$$OPT - f\left(S_{i+1}\right) = \delta_{i+1} \leq \left(1 - \frac{1}{k}\right)^{i+1} OPT \leq e^{-(i+1)/k} OPT.$$

In particolare se $i + 1 = k$ abbiamo la tesi: $f\left(S_{i+1}\right) \geq \left(1 - \frac{1}{e}\right) OPT$. □

4.2 Un algoritmo greedy generale

L'algoritmo 4, presentato nella sezione precedente, non offre nessuna garanzia per le funzioni submodulari non monotone. Per questo è necessario un approccio leggermente diverso per il caso generale. A tale scopo presentiamo l'algoritmo 5, uno dei due algoritmi randomizzati introdotti in [4].

Assunzione: Per questa sezione supponiamo che in X ci siano $2k$ elementi "fittizi" che non cambiano il valore di f, cioè $\exists D \subseteq X$, $|D| = 2k$ tale che $\forall x \in D$ vale che $\forall S \subseteq X \setminus \{x\}$ $f(S) = f(S \cup \{x\})$. Questa assunzione non ci fa perdere generalità,

Algoritmo 5 Algoritmo greedy randomizzato per vincolo di cardinalità

1: $S_0 \leftarrow \emptyset$
2: **for** $i \in \{1, \ldots, k\}$ **do**
3: Scelgo M_i tale che $|M_i| = k$ e in modo che $\sum_{u \in M_i} (f(S_{i-1} \cup \{u_i\}) - f(S_{i-1}))$ sia massima tra i possibili $M_i \subseteq X \setminus S_{i-1}$
4: Prendo $u_i \in M_i$ in modo uniformemente aleatorio
5: $S_i \leftarrow S_{i-1} \cup \{u_i\}$
6: **end for**
7: **return** $f(S_k)$

dato che se questi elementi non esistono, possiamo espandere X aggiungendo tali elementi fittizi.

Questa assunzione garantisce che $\sum_{u \in M_i} (f(S_{i-1} \cup \{u_i\}) - f(S_{i-1}))$ è una somma di termini non negativi.

Mostriamo inizialmente che se la funzione f è monotona allora questo algoritmo garantisce lo stesso fattore di approssimazione dell'algoritmo 4, cioè $\left(1 - \frac{1}{e}\right)$. Questo vuol dire che il presente algoritmo 5 è in un certo senso una generalizzazione randomizzata dell'algoritmo 4.

Proposizione 4.2. *Sia $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ funzione submodulare monotona, allora l'insieme S_k, così come calcolato nell'esecuzione dell'algoritmo 5, è tale che:*

$$\mathbb{E}[f(S_k)] \geq \left(1 - \frac{1}{e}\right) OPT.$$

Dimostrazione. Fissiamo un qualunque $i \in \{1, \ldots, k\}$. Supponiamo di fissare l'insieme S_{i-1}, condizionando le probabilità e i valori attesi seguenti rispetto a tale evento. Allora espandendo il valore assoluto:

$$\begin{aligned} k \cdot \mathbb{E}[f(S_i) - f(S_{i-1})] &= \cancel{k} \cdot \sum_{u \in M_i} \frac{1}{\cancel{k}} (f(S_{i-1} \cup \{u\}) - f(S_{i-1})) \\ &\geq \sum_{u \in S^* \setminus S_{i-1}} f(S_{i-1} \cup \{u\}) - f(S_{i-1}) \\ &\geq f(S_{i-1} \cup S^*) - f(S_{i-1}) \geq f(S^*) - f(S_{i-1}) = OPT - f(S_{i-1}). \end{aligned}$$

Qui la seconda disuguaglianza è vera perché $|S^* \setminus S_{i-1}| \leq k$ e M_i è scelto in modo da massimizzare $\sum_{u \in M_i} f(S_{i-1} \cup \{u\}) - f(S_{i-1})$; la terza disuguaglianza è vera per submodularità; l'ultima è vera per la monotonia di f. Il valore atteso è superfluo nelle ultime relazioni, dato che per ora abbiamo supposto di fissare S_{i-1}.

Se ora consideriamo di nuovo S_{i-1} come insieme aleatorio, cioè non più fissato, otteniamo:

$$\mathbb{E}[f(S^*) - f(S_{i-1})] \leq k \cdot \mathbb{E}[f(S_i) - f(S_{i-1})] \implies \delta_{i-1} \leq k(\delta_{i-1} - \delta_i),$$

dove abbiamo definito $\delta_i := f(S^*) - \mathbb{E}[f(S_i)]$ in modo simile alla dimostrazione del Teorema 4.1. Si può quindi concludere la dimostrazione analogamente a quanto fatto per tale teorema. □

A questo punto per dimostrare il risultato sul fattore di approssimazione per funzioni submodulari generali, ci serviranno i due lemmi. Entrambi sono tratti da [4], come il teorema successivo.

Lemma 4.3. *Sia $f : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$. Sia $A \subseteq X$, definiamo anche $k = |A|$. Sia $A(p) \subseteq A$ un insieme aleatorio in cui $\forall x \in A\ \mathbb{P}(x \in A(p)) \leq p$ (non necessariamente indipendentemente). Allora $\mathbb{E}[f(A(p))] \geq (1-p)f(\emptyset)$.*

Dimostrazione. Supponiamo che gli elementi di A siano $a_1, \ldots, a_k$ e che siano ordinati in ordine decrescente di probabilità di essere in $A(p)$; cioè le probabilità $p_i := \mathbb{P}(a_i \in A(p))$ per $i = 1, \ldots, k$ sono tali che $p \geq p_1 \geq p_2 \geq \cdots \geq p_k$. Chiamiamo anche gli insiemi $A_i := \{a_1, \ldots, a_i\}\ \forall i \in \{1, \ldots, k\}$ e $A_0 := \emptyset$. Quindi:

$$\begin{aligned}
\mathbb{E}[f(A(p))] &= \mathbb{E}\left[f(\emptyset) + \sum_{i=1}^{k} \mathbb{1}(a_i \in A(p))\,(f(A(p) \cap A_i) - f(A(p) \cap A_{i-1}))\right] \\
&\geq \mathbb{E}\left[f(\emptyset) + \sum_{i=1}^{k} \mathbb{1}(a_i \in A(p))\,(f(A_i) - f(A_{i-1}))\right] \\
&= f(\emptyset) + \sum_{i=1}^{k} \mathbb{E}[\mathbb{1}(a_i \in A(p))]\,(f(A_i) - f(A_{i-1})) \\
&= f(\emptyset) + \sum_{i=1}^{k} p_i\,(f(A_i) - f(A_{i-1})) \\
&= (1-p_1)f(\emptyset) + \sum_{i=1}^{k-1}(p_i - p_{i+1})f(A_i) + p_k f(A) \geq (1-p)f(\emptyset).
\end{aligned}$$

La prima disuguaglianza è giustificata dalla submodularità di f. La seconda è vera perché abbiamo tolto una somma di addendi non negativi al membro di destra, e $1 - p_1 \geq 1 - p$ dato che $p \geq p_1$.

Le uguaglianze sono, nell'ordine: una riscrittura con somme telescopiche, un'applicazione della linearità del valore atteso, una valutazione del valore atteso della funzione indicatrice $\mathbb{E}[\mathbb{1}(a_i \in A(p))] = p_i$ e una reindicizzazione della somma. □

Lemma 4.4. *$\forall i \in \{1, \ldots, k\}$, si ha che $\mathbb{E}[f(S^* \cup S_i)] \geq \left(1 - \frac{1}{k}\right)^i OPT$.*

Dimostrazione. All'i-esima iterazione dell'algoritmo 5 si ha che ogni elemento $x \in X \setminus S_{i-1}$ ha probabilità almeno $1 - \frac{1}{k}$ di non essere in S_i, infatti se $x \in M_i$ allora questa probabilità è $1 - \frac{1}{k}$ e altrimenti 1. Si vede facilmente per induzione allora che ogni elemento di X ha probabilità almeno $\left(1 - \frac{1}{k}\right)^i$ di non essere in S_i.

Definiamo ora la funzione submodulare $g : \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ come $g(S) := f(S^* \cup S)$. Su g useremo allora il lemma precedente, considerando S_i come insieme di elementi casuali presi con probabilità al più $1 - \left(1 - \frac{1}{k}\right)^i$. In questo modo concludo facilmente:

$$\mathbb{E}\left[f(S^* \cup S_i)\right] = \mathbb{E}\left[g(S_i)\right] \geq \left(1 - \frac{1}{k}\right)^i g(\emptyset) = \left(1 - \frac{1}{k}\right)^i f(S^*) = \left(1 - \frac{1}{k}\right)^i OPT.$$

□

Teorema 4.5. *Per una qualunque funzione submodulare $f : \mathbb{P}(X) \to \mathbb{R}_{\geq 0}$, l'algoritmo 5 ha un fattore di approssimazione atteso di almeno $\frac{1}{e}$, cioè:*

$$\mathbb{E}\left[f(S_k)\right] \geq \frac{1}{e} OPT.$$

Dimostrazione. Procediamo inizialmente in modo simile alla Proposizione 4.2. Fissiamo un $i \in \{1, \ldots, k\}$, e condizioniamo i valori attesi rispetto all'evento di avere S_{i-1} un certo insieme fissato. Quindi

$$\begin{aligned}\mathbb{E}\left[f(S_i) - f(S_{i-1})\right] &= \sum_{u \in M_i} \frac{f(S_{i-1} \cup \{u\}) - f(S_{i-1})}{k} \\ &\geq \sum_{u \in S^* \setminus S_{i-1}} \frac{f(S_{i-1} \cup \{u\}) - f(S_{i-1})}{k} \quad \geq \frac{f(S^* \cup S_{i-1}) - f(S_{i-1})}{k}\end{aligned}$$

espandendo il valore atteso e quindi usando la proprietà di massimalità con cui è stato scelto M_i e poi la submodularità.

Rilassiamo ora il condizionamento rispetto al valore di S_{i-1}, che fino ad ora era fissato. Prendiamo allora il valore atteso su tutti i possibili S_{i-1}, ed utilizzando la linearità del valore atteso e il lemma precedente troviamo:

$$\begin{aligned}\mathbb{E}\left[f(S_i) - f(S_{i-1})\right] &\geq \frac{\mathbb{E}\left[f(S^* \cup S_{i-1}) - f(S_{i-1})\right]}{k} \\ &\geq \frac{(1 - \frac{1}{k})^{i-1} OPT - \mathbb{E}\left[f(S_{i-1})\right]}{k}. \qquad (4.1)\end{aligned}$$

Ora voglio provare che:

$$\mathbb{E}[f(S_i)] \geq \frac{i}{k} \left(1 - \frac{1}{k}\right)^{i-1} OPT. \qquad (4.2)$$

Per induzione: il caso $i = 0$ è ovvio per la non negatività di f. Per il caso $i \geq 1$: supponiamo che la disuguaglianza sia vera per $i - 1$, quindi usando prima la

disuguaglianza 4.1 e poi l'ipotesi induttiva:

$$\begin{aligned}
\mathbb{E}\left[f(S_i)\right] &= \mathbb{E}\left[f(S_{i-1})\right] + \mathbb{E}\left[f(S_i) - f(S_{i-1})\right] \\
&\geq \mathbb{E}\left[f(S_{i-1})\right] + \frac{\left(1-\frac{1}{k}\right)^{i-1} OPT - \mathbb{E}\left[f(S_{i-1})\right]}{k} \\
&= \frac{1}{k}\left(1-\frac{1}{k}\right)^{i-1} OPT + \left(1-\frac{1}{k}\right)\mathbb{E}\left[f(S_{i-1})\right] \\
&\geq \frac{1}{k}\left(1-\frac{1}{k}\right)^{i-1} OPT + \left(1-\frac{1}{k}\right)\frac{i-1}{k}\left(1-\frac{1}{k}\right)^{i-2} OPT \\
&= \left(\frac{1}{k} + \frac{i-1}{k}\right)\left(1-\frac{1}{k}\right)^{i-1} OPT = \frac{i}{k}\left(1-\frac{1}{k}\right)^{i-1} OPT.
\end{aligned}$$

Le uguaglianze sono semplici calcoli o raccoglimenti di fattori comuni. Si è così dimostrata la relazione 4.2.

A questo punto è facile concludere, ponendo $i = k$ si ha:

$$\mathbb{E}[f(S_k)] \geq \frac{k}{k}\left(1-\frac{1}{k}\right)^{k-1} OPT \geq \frac{1}{e} OPT. \qquad \square$$

Nello stesso lavoro in cui è presentato l'algoritmo 5, è mostrato un secondo algoritmo che garantisce un fattore di approssimazione di almeno $\frac{1}{e} + 0.004$ [4], un valore strettamente maggiore di quello appena dimostrato. Questo risultato è però abbastanza complesso, e quindi oltre gli scopi di questo lavoro. Ad ogni modo tale risultato è notevole, dato che prova che $\frac{1}{e}$ non è il miglior fattore di approssimazione ottenibile in tempo polinomiale per il problema di massimizzazione di funzioni submodulari con vincolo di cardinalità.

Bibliografia

[1] Shabbir Ahmed and Alper Atamtürk. Maximizing a class of submodular utility functions. *Mathematical programming*, 128(1):149–169, 2011.

[2] Francis Bach et al. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in machine learning*, 6(2-3):145–373, 2013.

[3] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 649–658, Oct 2012.

[4] Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Submodular Maximization with Cardinality Constraints. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1433–1452. Society for Industrial and Applied Mathematics, January 2014.

[5] William H. Cunningham. On submodular function minimization. *Combinatorica*, 5(3):185–192, Sep 1985.

[6] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, July 1998.

[7] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[8] Michel Goemans. Chernoff bounds, and some applications. 2014.

[9] Sudipto Guha and Samir Khuller. Greedy strikes back: improved facility location algorithms. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98, page 649–657, USA, 1998. Society for Industrial and Applied Mathematics.

[10] Andreas Krause and Daniel Golovin. Submodular Function Maximization. In Lucas Bordeaux, Youssef Hamadi, and Pushmeet Kohli, editors, *Tractability*, pages 71–104. Cambridge University Press, 1 edition, February 2014.

[11] George Nemhauser, Laurence Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14:265–294, 12 1978.

[12] Vijay V. Vazirani. *Approximation Algorithms*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

Appendice A

Chernoff bounds

In questa appendice riportiamo alcuni risultati riguardo a stime sulla misura di probabilità delle code di una somma di distribuzioni di Bernoulli, riportate da [8].

Cominciamo da un teorema elementare:

Teorema A.1 (Disuguaglianza di Markov)**.** *Sia X una variabile aleatoria discreta reale non negativa. Allora vale:*

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}[X]}{a} \qquad \forall a > 0.$$

Dimostrazione. Sia $A \subseteq \mathbb{R}$ l'insieme discreto contenente i valori che X può assumere. Chiamiamo $p_X : A \to \mathbb{R}_{\geq 0}$ la densità di probabilità di X:

$$\mathbb{E}[X] = \sum_{x \in A} x p_X(x) dx \geq \mathbb{P}(X \geq a) \cdot a. \qquad \square$$

A partire da questo risultato si possono dimostrare le seguenti disuguaglianze, nella forma usata nel Capitolo 3:

Teorema A.2 (Chernoff bounds)**.** *Siano $X_1, \ldots, X_n$ v.a. indipendenti e identicamente distribuite (i.i.d.) con distribuzione di Bernoulli di parametro $p \in [0, 1]$.*

Sia $X = \sum_{i=1}^{n} X_i$, e sia $\mu = \mathbb{E}[X] = np$. Allora $\forall \delta \in (0, 1)$ valgono:

$$\mathbb{P}\left(X \geq (1+\delta)\mu\right) \leq e^{-\delta^2 \mu/3} \qquad e \qquad \mathbb{P}\left(X \leq (1-\delta)\mu\right) \leq e^{-\delta^2 \mu/3}.$$

Inoltre $\mathbb{P}\left(|X - \mu| \geq \delta\mu\right) \leq 2e^{-\delta^2 \mu/3}$.

Dimostrazione. Sia $M_X : \mathbb{R} \to \mathbb{R}$, $M_X(t) = \mathbb{E}\left[e^{tX}\right]$ la funzione generatrice dei momenti di X. Allora per questa vale:

$$M_X(t) = \prod_{i=1}^{n} M_{X_i}(t) = \left(pe^t + 1 - p\right)^n = \left(1 + p(e^t - 1)\right)^n = e^{(e^t - 1)np} = e^{(e^t - 1)\mu}.$$

Si noti che la prima uguaglianza è vera grazie all'indipendenza delle v.a. $X_1, \ldots, X_n$.

Fisso ora $\delta \in (0,1)$. Con la disuguaglianza di Markov troviamo:

$$\mathbb{P}(X \geq (1+\delta)\mu) = \mathbb{P}(e^{sX} \geq e^{s(1+\delta)\mu}) \leq M_X(s)e^{-s(1+\delta)\mu} = e^{(e^s-1)\mu - s(1+\delta)\mu}$$

dove $s \in \mathbb{R}_{\geq 0}$ è un parametro libero. Per minimizzare il membro di destra scelgo $s = \log(1+\delta)$. Infatti l'esponenziale è monotono, e per minimizzarlo è sufficiente minimizzare l'esponente, di cui osserviamo la derivata: $\frac{\partial}{\partial s}((e^s-1)\mu - s(1+\delta)\mu) = \mu(e^s - 1 - \delta)$. Questa si annulla solo per $s = \log(1+\delta)$, punto in cui deve allora essere minimo.

Assegnando questo valore ad s la disuguaglianza diventa:

$$\mathbb{P}\left(X \geq (1+\delta)\mu\right) \leq e^{(\delta - (1+\delta)\log(1+\delta))\mu}.$$

Allo stesso modo troviamo la seguente stima per la coda di sinistra:

$$\begin{aligned}\mathbb{P}\left(X \leq (1-\delta)\mu\right) = \mathbb{P}\left(e^{-sX} \geq e^{-s(1-\delta)\mu}\right) &\leq M_X(-s)e^{s(1-\delta)\mu} = e^{(e^{-s}-1)\mu + s(1-\delta)\mu} \\ &\leq e^{(e^{-s}-1+s(1-\delta))\mu}.\end{aligned}$$

In questo caso la derivata dell'esponente rispetto a s è $\mu(-e^{-s} + (1-\delta))$, quindi $s = -\log(1-\delta)$ minimizza il membro di destra. Da cui:

$$\mathbb{P}\left(X \leq (1-\delta)\mu\right) \leq e^{(-\delta - (1-\delta)\log(1-\delta))\mu}.$$

Ora per entrambe le code possiamo limitare superiormente l'esponente del membro di destra con $-\frac{\delta^2}{3}\mu$, infatti:

$$(\delta - (1+\delta)\log(1+\delta))\mu \leq \left(\delta - (1+\delta)\frac{2\delta}{2+\delta}\right)\mu = \frac{(2+\delta) - 2(1+\delta)}{2+\delta}\delta\mu = -\frac{\delta^2\mu}{2+\delta};$$

$$(-\delta - (1-\delta)\log(1-\delta))\,\mu \leq \left(-\delta - (1-\delta)\left(-\delta + \frac{\delta^2}{2}\right)\right)\mu = \frac{-3\delta^2 + \delta^3}{2}\mu \leq -\frac{\delta^2\mu}{2}.$$

In queste stime si è usato: nel primo caso che $\log(1+x) \leq \frac{2x}{2+x}$ che vale per $x \geq 0$, mentre per il secondo che $\log(1-x) \geq -x + \frac{x^2}{2}$ per $0 < x < 1$.

La tesi segue ora dal fatto che $\frac{\delta^2\mu}{2} \geq \frac{\delta^2\mu}{2+\delta} \geq \frac{\delta^2\mu}{3}$. Possiamo quindi stimare entrambe le code con $e^{-\frac{\delta^2\mu}{3}}$ come nella tesi.

La seconda parte della tesi è una semplice unione di probabilità:

$$\begin{aligned}\mathbb{P}\left(|X-\mu| \geq \delta\mu\right) &= \mathbb{P}\left(X \geq (1+\delta)\mu \vee X \leq (1-\delta)\mu\right) \\ &\leq \mathbb{P}\left(X \geq (1+\delta)\mu\right) + \mathbb{P}\left(X \leq (1-\delta)\mu\right) \leq 2e^{-\delta^2\mu/3}. \qquad \square\end{aligned}$$