Metodi per l'Allocazione Ottima di Risorse

Dott.ssa Maria Silvia Pini

Dipartimento di Matematica Pura e Applicata
Email: mpini@math.unipd.it
Resp. accademico: Prof.ssa Francesca Rossi

1 Introduzione

In problemi di allocazione di commesse tra le aziende che fanno parte di una rete succede spesso che le singole aziende e i clienti abbiano criteri di interessi che sono spesso in conflitto tra di loro. In questi scenari è fondamentale capire come allocare queste commesse in maniera ottima in modo tale che nessuna azienda venga sfavorita e il cliente sia soddisfatto. Motivati dalla necessità di trovare una metodologia per l'allocazione ottima delle commesse in contesti multi-aziendali, in questo documento analizzaremo i principali metodi che sono presenti nella letteratura per risolvere questo tipo di problemi.

In questi problemi, come in altri problemi reali [11,28,16,25], risulta necessario trovare una soluzione che sia ottima contemporaneamente per i diversi criteri di interesse (anche chiamati 'obiettivi') che spesso si trovano in competizione tra di loro. In questo caso si parla di problema di ottimizzazione multi-obiettivo.

In genere, quando si affronta questo tipo di problema, la soluzione ottimale ottenuta non è mai unica (come nel caso di ottimizzazione ad un singolo obiettivo), ma ci si trova di fronte ad un insieme di soluzioni ugualmente ottimali rispetto al problema dato. Tale insieme di soluzioni prende il nome di insieme delle soluzioni Pareto-ottime. Spetta poi alla figura del Decision Maker, cioè colui che prende la decisione, il compito di decidere quale delle soluzioni trovate all'interno dell'insieme delle soluzioni Pareto-ottime è quella più adatta a soddisfare le sue esigenze (il che può significare privilegiare un criterio piuttosto che un altro, o scegliere un valore intermedio in modo che tutti gli criteri siano ottimizzati allo stesso modo, e così via).

Un valido strumento per la risoluzione di questi problemi di ottimizzazione a più obiettivi sono gli algoritmi evolutivi [10,12,1,11,26]. Tali algoritmi consistono in una classe di metodi di ottimizzazione che sono particolarmente adatti per approssimare l'insieme delle soluzioni

Pareto-ottime, poichè lavorano con popolazioni di soluzioni e non con una singola candidata soluzione. Questi algoritmi sono capaci di approssimare parecchi membri dell'insieme delle soluzioni Pareto-ottime dopo una sola esecuzione. In questo documento vedremo alcuni dei più importanti algoritmi evolutivi che sono stati definiti in lettereratura. Oltre agli algoritmi evolutivi, descriveremo anche altri metodi di soluzione che hanno avuto successo in letteratura che sono i metodi che si basano sulle preferenze [21,14].

Il documento è organizzato come segue. Nella Sezione 2 descriviamo i problemi di ottimizzazione multi-obiettivo e presentiamo alcuni degli algoritmi più importanti noti in letteratura per risolverli. Nella Sezione 3 analizziamo il concetto della fairness (cioè dell'equità) nell'allocazione di risorse. Nella Sezione 4 presentiamo altri risultati noti in letteratura nell'ambito dell'ottimizzazione multi-obiettivo e nell'ambito della allocazione fair di risorse. Infine, nella Sezione 5 riassumiamo il contenuto del documento e presentiamo alcune idee interessanti che intendiamo considerare per risolvere nel modo migliore il problema dell'allocazione delle commesse tra le aziende di una rete.

2 Algoritmi per l'ottimizzazione multi-obiettivo

In molti problemi reali viene richiesta la simultanea ottimizzazione di una serie di obiettivi che talvolta sono in competizione tra di loro. Una soluzione ottimale rispetto ad un obiettivo richiede allora un compromesso con gli altri obiettivi.

Quando un problema di ottimizzazione richiede la presenza di più di una funzione obiettivo, il problema di trovare una o più soluzioni ottimali prende il nome di ottimizzazione multi-obiettivo.

Ci sono vari approcci in letteratura per risolvere problemi di ottimizzazione multi-obiettivo che sono riassunti in [9]. Quelli che analizzeremo in questa sezione sono quelli che si basano sul concetto di dominanza.

2.1 La frontiera Pareto-ottima

Nei problemi di ottimizzazione multi-obiettivo ogni soluzione è definita da un vettore di variabili decisionali $x = (x_1, \ldots x_n)$. Si assume che ci siano M funzioni obiettivo $f_i(x)$, dove $i = 1, \ldots, M$. Le funzioni obiettivo sono una rappresentazione matematica dei criteri secondo i quali si vuole ottimizzare. Spesso questi criteri sono in conflitto gli uni con gli altri [25].

Vogliamo trovare un insieme di valori per le variabili decisionali che ottimizza un insieme di funzioni obiettivo.

Diciamo che un vettore di decisione x *domina* un vettore di decisione y se e solo valgono le condizioni seguenti:

- $f_i(x) \geq f_i(y)$, per ogni obiettivo $i \in 1, \ldots, M$,
- $f_i(x) > f_i(y)$, per qualche obiettivo $i \in 1, \ldots, M$.

Quindi il vettore di decisione x domina il vettore di decisione y se e solo x è migliore o uguale a y secondo tutti i criteri e per un criterio x è strettamento meglio di y.

Tutti i vettori di decisione che non sono dominati da nessun altro vettore di decisione vengono detti *non-dominati*, oppure *Pareto-ottimi*, e costituiscono la frontiera Pareto-ottima. Queste sono le soluzioni per le quali nessun obiettivo può essere migliorato senza peggiorarne un altro.

Lo scopo dell'ottimizzazione multi-obiettivo è quello di trovare un vettore ideale di variabili decisionali x, che ottimizza un vettore di M funzioni obiettivo $f_i(x)$, soggetto a vincoli di disuguaglianza $g_j(x) \geq 0$ e vincoli di uguaglianza $h_k(x) = 0$, dove $j = 1, 2, \ldots, J$ e $k = 1, 2, \ldots, K$.

Senza perdita di generalità, un problema di ottimizzazione multi-criterio, può pertanto essere definito come segue:

$$\begin{gathered} \text{Massimizzare } \{f_1(x), f_2(x), \ldots, f_M(x)\} \\ \text{soggetto a} \\ g_j(x) \geq 0, \text{ per } j = 1, 2, \ldots, J \\ h_k(x) = 0, \text{ per } k = 1, 2, \ldots, K, \end{gathered}$$

dove x è il vettore delle variabili decisionali, $f_i(x)$ è l'i-esima funzione obiettivo e $g_j(x)$ e $h_i(x)$ sono vincoli.

2.2 Gli algoritmi evolutivi

Le metaeuristiche sono una famiglia di tecniche di ottimizzazione approssimata per risolvere problemi combinatori che hanno ricevuto un crescente interesse in questi ultimi anni. Tra le metaeuristiche, gli algoritmi evolutivi sono particolarmente desiderabili per risolvere problemi di ottimizzazione multi-criterio per la loro natura basata sulla popolazione. Questo

permette loro di catturare le relazioni di dominanza nella popolazione che servono come guida per la ricerca verso la frontiera Pareto-ottima.

Questi algoritmi trattano simultaneamente con un insieme di soluzioni possibili (la cosiddetta popolazione) e possono trovare buone approssimazioni dell'insieme Pareto-ottimo. In particolare, grazie alla loro capacità di esplorare le similarità delle soluzioni, tali algoritmi riescono ad approssimare bene la frontiera Pareto-ottima dopo una singola esecuzione. Questo fatto ha comportato un utilizzo degli algoritmi evolutivi multi-obiettivo in sempre più numerose applicazioni.

Poichè gli algoritmi evolutivi non sono esatti, diverse esecuzioni possono produrre soluzioni diverse, quindi si devono effettuare diverse esecuzioni dello stesso algoritmo su un dato problema per descrivere la performance dell'algoritmo su quel problema.

Una descrizione dettagliata degli algoritmi multi-obiettivo evolutivi viene presentata in [8,7,10].

L'algoritmo NSGA-II. Uno dei migliori algortmi evolutivi noti in letteratura è l'algoritmo NSGA-II. Questo algoritmo è un algoritmo genetico multi-obiettivo [12]. La sigla NSGA-II vuol dire non-dominated sorting genetic algorithm II, per sottolineare il fatto che in esso le soluzioni del problema di ottimizzazione in questione vengono ordinate in accordo al concetto di non-dominanza.

Nell'NSGA-II la popolazione ha bisogno di essere ordinata in accordo ad un ordine ascendente di non-dominanza. Più precisamente, l'algoritmo opera una classificazione dell'intera popolazione, suddividendola in classi, o rank. Le migliori soluzioni non dominate sono chiamate soluzioni non-dominate di rank 1, ovvero una volta trovati tutti gli elementi non-dominati di una popolazione ad una determinata generazione, ad essi viene assegnata un rank pari ad 1. Tali elementi verranno poi momentaneamente trascurati per fare in modo che lo stesso procedimento venga applicato agli individui rimanenti, così da determinare il secondo rank, poi il terzo, e così via. Alla fine del processo ogni individuo apparterrà ad un rank. La complessità di questa procedura è la somma delle complessità richieste nell'identificazione di ogni insieme non dominato. È importante notare che una volta identificato il primo insieme non-dominato, il numero di soluzioni rimanenti è minore del numero originale di individui della popolazione, quindi le classificazioni successive alla prima richiederanno una complessità computazionale minore.

Nell'algoritmo NSGA-II il processo di classificazione ha complessità O(MN^2), dove M è il numero di obiettivi ed N è la grandezza della popolazione. In questo algoritmo la popolazione viene ordinata secondo la relazione della non-dominanza in parecchie frontiere. In questo modo, alle soluzioni nelle frontiere migliori vengono dati valori, detti valori di fitness, più alti.

L'Algoritmo NSGA-II usa una misura, chiamata crowding distance, per fornire una stima della densità delle soluzioni che appartengono alla stessa frontiera. Questo parametro viene usato per favorire la diversità all'interno della popolazione. A soluzioni con un'alta crowding distance viene assegnato un più alto valore di fitness di quelle con una più bassa crowding distance.

Deb et al. [12] assumono che ogni individuo i della popolazione abbia due attributi: il rank di non-dominanza (che indichiamo con i_{rank}) e la crowding distance (che indichiamo con $i_{distance}$). Dati due individui della popolazione i e j, diciamo che i *è meglio di* j se

- $i_{rank} < j_{rank}$ oppure
- $i_{rank} = j_{rank}$ e $i_{distance} > j_{distance}$.

Pertanto, tra due soluzioni con differenti rank di non-dominanza, la soluzione con il rank più basso è quella più preferita, e tra due soluzioni che appartengono alla stessa frontiera, quella più preferita è la soluzione collocata nella regione meno affollata.

Una descrizione dettagliata dell'algoritmo NSGA-II è presentata in [12].

L'algoritmo Two-Archive. Un altro algoritmo multi-obiettivo evolutivo noto in letteratura è l'algoritmo Two-Archive [26]. In questo algoritmo ci sono due archivi che vengono utilizzati per collezionare le soluzioni candidate della popolazione: l'archivio di convergenza (AC) e l'archivio di diversità (AD). Se una soluzione non-dominata selezionata dalla popolazione domina altri membri negli archivi, allora entra nell'archivio AC e i membri dominati vengono eliminati, altrimenti viene inserita nell'archivio AD senza eliminare nessun altro elemento degli archivi.

La taglia totale degli archivi è fissata, ma la taglia di ogni archivio varia. Quando il numero dei membri negli archivi supera la capacità degli archivi, i membri dell'archivio AD con la minima distanza dai membri dell'archivio AC sono eliminati finchè la taglia totale degli archivi sta sotto la soglia.

Maggiori dettagli relativi a questi algoritmo sono presentati in [26].

2.3 Metodi basati sulle preferenze

Un'altra tipologia di algoritmi che sono stati usati con successo per risolvere problemi di ottimizzazione multi-criterio sono gli algoritmi di ottimizzazione che si basano sulle preferenze. Questi algoritmi usano le preferenze per guidare la ricerca delle soluzioni ottime [21].

In molti problemi reali, come ad esempio in problemi di configurazione, capita spesso che ci siano pochi vincoli e quindi tante soluzioni sono ammissibili. Pertanto è necessario definire un meccanismo per trovare le soluzioni più preferite. Esprimere le preferenze sulle possibili decisioni o sui criteri di ottimizzazione da adottare è una tecnica molto utile per individuare tra tutte le soluzioni possibili quelle effettivamente interessanti.

Pensiamo ad esempio ad un sistema automatizzato per la pianificazione delle vacanze, che sceglie una o più destinazioni per la vacanza da un catalogo molto grande. Gli utenti esprimono le loro richieste, come ad esempio le attività che vorrebbero fare in vacanza, come ad esempio andare in canoa e fare surf. Poi ci sono dei vincoli di compatibilità tra le varie destinazioni che possono essere visitate nella medesima vacanza e dei vincoli globali di risorse, come ad esempio il prezzo della vacanza. In questi problemi c'è un numero grandissimo di soluzioni tutte ammissibili. Esprimere le preferenze sulle varie destinazioni è un buon modo per selezionare solo alcune delle soluzioni interessanti tra tutte quelle ammissibili. Per esempio, l'utente può preferire le Hawaii alla Florida per fare surf. Inoltre l'utente può avere delle preferenze sui criteri generali di interesse. Per esempio, può preferire una vacanza poco costosa. Inoltre può preferire una vacanza di buona qualità e con poca lontananza tra le varie destinazioni. Ecco allora che siamo di fronte ad un problema in cui ci sono vari criteri da ottimizzare: il prezzo e la distanza devono essere il più bassi possibile e la qualità deve essere la più alta possibile.

Nell'ambito dell'Intelligenza Artificiale sono stati sviluppati diversi metodi per rappresentare e trattare le preferenze [6,17,4,20]. Se applicassimo per esempio l'approccio descritto in [22] per trattare il nostro problema multi-obiettivo, il sistema potrebbe proporre una vacanza poco costosa ma di scarsa qualità oppure una vacanza molto costosa e di qualità buona, comunque non proporrebbe nessuna soluzione che cerchi di bilanciare i due criteri del costo e della qualità. Per evitare questo problema, in [21] il metodo di ricerca basato sulle preferenze proposto in [20] è stato generalizzato per poter gestire anche l'ottimizzazione multi-criterio.

Nell'ambito dell'ottimizzazione multi-criterio sono stati definite varie nozioni di ottimalità:

- la Pareto ottimalità;
- l'ottimalità lessicografica;
- l'ottimalità rispetto a somme pesate.

La Pareto ottimilità l'abbiamo descritta all'inizio di questa sezione. Ricordiamo che le soluzioni Pareto-ottime sono le soluzioni non-dominate. L'ottimalità lessicografica consiste nell'ordinare le soluzioni sulla base di un ordinamento lessicografico, che dipende da un certo ordine di importanza dei criteri, e nel selezionare la soluzione migliore rispetto a questo ordinamento. Per esempio, assumiamo che ci siano due criteri, c_1 e c_2, e che c_1 sia più importante di c_2. Allora, per selezionare la soluzioni migliore, dobbiamo prima individuare quelle che hanno il valore più alto rispetto a c_1 e poi tra queste dobbiamo prenderne una di quelle che ha il valore più alto rispetto a c_2. L'ottimalità rispetto a somme pesate consiste invece nel considerare un unico obiettivo dove sono presenti tutti i criteri con uguale o diverso peso.

Un descrizione approfondita dei vari metodi multi-obiettivo e delle varie nozioni di ottimalità viene presentata in [15]. Sulla base di questi metodi si possono determinare due tipi di soluzioni:

- le soluzioni estreme, cioè le soluzioni in cui un criterio è favorito rispetto agli altri;
- le soluzioni bilanciate, cioè le soluzioni in cui i vari criteri sono favoriti nel modo più simile possibile.

Le soluzioni bilanciate, che rappresentano una sorta di compromesso tra i vari criteri, richiedono che i criteri siano confrontabili. Questo bilanciamento viene raggiunto solitamente utilizzando un metodo di standardizzazione. Queste soluzioni bilanciate non vengono ottenute usando l'approccio delle somme pesate, mentre vengono ottenute adottando il nuovo approccio lessicografico descritto in [14]. Ritornando al nostro esempio delle vacanze, secondo questo approccio dobbiamo procedere come segue per trovare un compromesso tra un buon prezzo e una buona qualità: dobbiamo prima minimizzare il massimo tra (le versioni standardizzate) di prezzo e qualità, fissare un criterio (per esempio, la qualità standardizzata) al risultante minimo e poi minimizzare l'altro criterio (cioè il prezzo).

In [21] viene presentata una procedura di ricerca basata sulle preferenze per trovare una variante delle soluzioni bilanciate descritte in [14] che permette di bilanciare solo alcuni

gruppi di criteri e non tutti. Nel nostro esempio, potremmo voler bilanciare solo i criteri legati al prezzo e alla qualità e non quello relativo alla distanza.

Il metodo presentato in [21] è un'estensione di quello presentato in [20] che permette di esprimere questi due tipi di preferenze:

- le preferenze sui criteri: per ogni criterio, si considera un ordinamento parziale stretto tra i possibili valori e si seleziona il miglior valore rispetto a quell'ordinamento;
- le preferenze tra i criteri: se un criterio c_1 è più importante di un criterio c_2, allora ogni assegnamento di valori a c_1 è più importante di ogni assegnamento di valori a c_2.

Tale metodo funziona come segue: seleziona i criteri sulla base dei compromessi tra i criteri e sulla base delle preferenze e poi li passa come obiettivo da ottimizzare ad un sottoproblema che viene risolto usando una ricerca Branch and Bound che si basa sui vincoli. Pertanto esegue una sequenza di sottoproblemi di minimizzazione con funzioni obiettivo che cambiano, cioè con criteri di ottimizzazione diversi.

Una descrizione dettagliata di questa procedura viene presentata in [21].

3 L'equità nell'allocazione

Nella sezione precedente abbiamo esaminato i metodi che hanno avuto più successo in letteratura nell'ambito dell'ottimizzazione multi-criterio, che è un aspetto cruciale nei problemi di allocazione di risorse. Ora analizzeremo un altro aspetto che è molto importante in questi problemi: il concetto dell'equità (anche chiamata 'fairness') dell'allocazione.

Questo aspetto è stato studiato in [16] nell'ambito di problemi di requirements analysis. In questi problemi c'è un sistema software esistente e ci sono vari clienti che hanno delle richieste che vogliono che vengano sviluppate nella prossima edizione del software. Lo scopo è quello di allocare in maniera equa le risorse, cioè soddisfare in modo equo le richieste dei clienti in modo tale da non privilegiare nessun cliente rispetto agli altri.

Gli aspetti relativi alla fairness che sono stati analizzati in [16] sono i seguenti:

- a quale livello è possibile dire che un'allocazione è fair?
- quale è una ragionevole misura della fairness?

Questi due aspetti sono interrelati e complicati dal fatto che non c'è un unica nozione di fairness accettata da tutti gli utenti. Per esempio, nei problemi di requirements analysis, un'allocazione può essere fair se:

– soddisfa lo stesso numero di richieste per ogni cliente, oppure
– dà lo stesso livello di soddisfazione ad ogni cliente (poichè alcune richieste contano più di altre), oppure
– se dà uguale costo ad ogni cliente.

In [16] vengono introdotte delle tecniche per l'analisi dei compromessi tra le varie nozioni di fairness dei clienti, dove ci sono più clienti con richieste diverse e spesso conflittuali tra di loro. Più precisamene, viene adottata una ricerca Pareto ottima multi-obiettivo per l'allocazione ottima, che permette di trattare contemporaneamente le varie nozioni di fairness che ogni cliente ha in mente. Adottando questa procedura, è possibile individuare quelle soluzioni che bilanciano i compromessi tra le varie nozioni di fairness. Lo scopo dell'analisi presentata in [16] non è di dare un sistema automatico di allocazione, ma di dare dei suggerimenti al decision maker, che è la persona che prenderà la decisione, sui possibili modi di allocazione. Questa analisi è utile, in particolare, per:

– mostrare al decisore dove ci sono i potenziali problemi di bilanciamento tra le varie nozioni di fairness,
– permettere al decisore di mostrare al cliente che la soluzione adottata è fair, cioè giusta, secondo più criteri di fairness.

La ricerca automatizzata delle regioni ottime dello 'spazio della fairness' ha applicazioni in vari campi, quali la negoziazione, la mediazione e la risoluzione di conflitti, sia nei problemi di requirement analysis, sia in problemi generali di allocazione di risorse in cui ci sono più interessi che sono conflittuali tra di loro.

4 Articoli scientifici correlati

I problemi in cui vari utenti (aziende, clienti, ...) hanno punti di vista competitivi e spesso conflittuali sono stati esaminati in vari articoli scientifici. In [5] è stato proposto un modello, chiamato WinWin model, per aiutare il processo di negoziazione degli azionisti di una società sulla base dell'analisi delle preferenze multi-criterio. Un altro approccio che è stato definito per risolvere conflitti tra azionisti è l'approaccio ViewPoint [13]. Tale approccio separa le differenti opinioni tra gli azionisti ed è capace di individuare automaticamente i conflitti. Sempre nell'ambito della negoziazione è stato anche proposto un supporto automatizzato per risolvere i conflitti [27].

Inizialmente per risolvere i problemi in cui gli utenti avevano criteri di ottimizzazione differenti si è adottata una formulazione del problema ad un singolo obiettivo in cui i vari vincoli e i vari obiettivi che caratterizzavano il problema venivano combinati insieme in un'unica funzione obiettivo. In questo ambito è stata sviluppata una vasta gamma di algoritmi di ottimizzazione: algoritmi di programmazione lineare intera, algoritmi greedy, algoritmi di branch and bound, algoritmi di simulated annealing e algoritmi genetici [3,18]. Tuttavia la formulazione a un singolo obiettivo ha lo svantaggio che non si riescono a bilanciare i vari criteri.

Più recentemente ci sono stati molti lavori nel campo delle formulazioni multi-obiettivo [10,28]. In questo contesto ognuno degli obiettivi da ottimizzare viene trattato come fosse un goal separato. Questo permette di esplorare la frontiera Pareto ottima delle soluzioni non-dominate.

Per ridurre la regione di interesse della ricerca e per guidare la ricerca, sono stati sviluppate anche delle procedure che tengono conto delle preferenze espresse dagli utenti [11]. In particolare, in [11] viene proposto un metodo interattivo che ingloba le preferenze degli utenti. Tali preferenze vengono utilizzate per guidare la ricerca multi-obiettivo. L'idea è quella di ridurre lo spazio di ricerca focalizzandosi solo sulle regioni della frontiera Pareto più favorevoli. Questo metodo ha potenziali applicazioni in molti ambiti a condizione che l'utente sia disposto a fornire le sue preferenze durante la ricerca.

Altri metodi che si basano sul concetto delle preferenze per risolvere questo tipo di problemi sono presentati in [23,24]. In particolare, in [23] viene proposto un metodo basato su vincoli e preferenze che rende molto più trasparente all'utente l'intero processo di soluzione e indica i trade-off, cioè i compromessi, che sono stati fatti durante la ricerca. Questo permette all'utente di raffinare il modello di preferenza, ottenendo così il pieno controllo sul risolutore del problema. In [24] viene proposto un metodo che si basa sulle preferenze espresse nella forma dei vincoli soft [4] per trovare un sottinsieme delle soluzioni Pareto-ottime, chiamate soluzioni sorted-Pareto-ottime. Questo metodo permette di presentare al decision maker, cioè alla persona che prenderà la decisione, un insieme più piccolo di soluzioni ottime rispetto all'insieme delle soluzioni Pareto-ottime.

Nel contesto dell'allocazione di risorse, oltre alla nozione di ottimalità, è stato studiato anche il concetto della fairness [16], cioè a che livello è possibile dire che un'allocazione è fair e quale è un modo ragionevole di definire la fairness. Questo tema è stato studiato nell'ambito dei problemi di requirements analysis, dove ci sono più clienti con interessi competitivi e

spesso conflittuali. Si è mostrato che in genere non c'è mai un unica nozione di fairness, perchè quello che potrebbe essere giusto per un cliente potrebbe non esserlo per un altro. In [16] gli autori hanno mostrato che, usando una ricerca Pareto ottima multi-obiettivo per l'allocazione ottima, è possibile trattare contemporaneamente le nozioni di fairness che ogni cliente ha in mente ed è possibile individuare quelle soluzioni che bilanciano i compromessi tra le varie nozioni di fairness. Lo scopo di questa analisi non è quello di dare un sistema automatico di allocazione, ma è di fornire dei suggerimenti al decision maker sui possibili modi di allocazione. In particolare, è utile per mostrare al decisore dove ci sono potenziali problemi di bilanciamento tra le varie nozioni di fairness, e per permettere al decisore di mostrare al cliente che la soluzione adottata è fair, cioè giusta, secondo più criteri di fairness.

5 Conclusioni e direzioni di ricerca future

In questo documento abbiamo considerato il problema dell'allocazione di risorse tra utenti che hanno punti di vista competitivi e spesso conflittuali. In particolare, abbiamo descritto le varie nozioni di ottimalità che sono state definite in questi problemi e abbiamo presentato i metodi che hanno avuto più successo nella letteratura per trovare le soluzioni ottime secondo le varie nozioni di ottimalità. Inoltre, abbiamo esaminato il concetto della fairness nell'ambito dell'allocazione di risorse.

Come direzione di ricerca futura, per individuare la miglior strategia di allocazione delle commesse tra una rete di aziende, che hanno interessi spesso conflittuali, vogliamo esaminare i principali metodi di negoziazione che sono stati definiti in letteratura ed utilizzati in problemi reali [2,19]. Inoltre, vista l'importanza che hanno avuto le preferenze nella letteratura, vogliamo anche considerare la possibilità di elicitare le preferenze dalle aziende coinvolte nella rete. Possiamo elicitare due tipi di preferenze: le preferenze sul tipo di allocazione e le preferenze sui criteri di ottimizzazione. Per esempio, un'azienda può preferire prendere tante commesse di piccole dimensioni in tempi successivi, piuttosto che ricevere un'intera commessa di grandi dimensioni, perchè in questo modo è più sicura di realizzarla nel tempo stabilito, oppure perchè in questo modo il pagamento avviene prima. Altre aziende invece possono preferire grandi commesse per avere tanto lavoro subito. Oltre alle preferenze sui tipi di allocazioni, le aziende possono anche esprimere le loro preferenze sui criteri che si devono adottare per scegliere l'allocazione migliore delle commesse. Per esempio, alcune aziende possono voler massimizzare il guadagno, altre aziende invece possono voler minimizzare i

costi, e così via. Le preferenze sui criteri espresse dalle varie aziende possono essere ad esempio utilizzate per generare un ordinamento dei criteri di ottimizzazione che ci servirà per scegliere la soluzione migliore o un sottinsieme delle soluzioni migliori. Per generare questo ordinamento possiamo poi utilizzare una regola di voto che può essere accettata per esempio dalla maggioranza delle aziende coinvolte nella rete.

Riferimenti bibliografici

1. pages –.
2. R. Aydogan and P. Yolum. Effective negotiation with partial preference information. In *Proceedings of AAMAS'10*, pages 1605–1606, 2010.
3. A. J. Bagnall, V. J. Rayward-Smith, and I. Whittley. The next release problem. *Information & Software Technology*, 43(14):883–890, 2001.
4. S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint satisfaction and optimization. *Journal of ACM*, 44(2):201–236, 1997.
5. H. Boehm, T. Rodgers, and M. Deutsch. Appling winwin to quality requirements: a case study. In *Proceedings of IEEE Conference on Software Engeenering*, pages 555–564, 2001.
6. G. Brewka. Preferred subtheories: An extended logical framework for default reasoning. In *Proceedings of IJCAI'89*, pages 1043–1048, 1989.
7. C. A. Coello Coello. Evolutionary multiobjective optimization: a historical view of the field. *IEEE Computational Intelligent Magazine*, 1.
8. C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont. *Evolutionary algorithms for solving multi-objective problems*. Kluwer, New-York, 2002.
9. Y. Collette and P. Siarry. *Multi-objective optimization: principles and case studies*. Springer, Berlin, 2004.
10. K. Deb. *Multi-objective optimization using evolutionary algorithms*. Wiley, Chichester, 2001.
11. K. Deb and A. Kumar. Interactive evolutionary multi-objective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'07)*, pages 781–788, 2007.
12. K. Deb, A. Pratap, S. Argawal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transaction of Evolutionary Computation*, 6(2):182–197, 2002.
13. S. Easterbrook and M. Chechik. A framework for multi-valued reasoning over inconsistent viewpoints. In *Proceedings of International Conference of Software Engeenering*, pages 411–420, 2001.
14. M. Ehrgott. A characterization of lexicographic max-ordering solutions. In *Methods of Multicriteria Decision Theory, Proceedings of the 6th Workshop of the DGOR-Working Group Multicriteria Optimization and Decision Theory*, pages 193–202, 1997.

15. M. Ehrgott. Approximation algorithms for combinatorial multicriteria optimization problems. *International Transactions in Operational Research*, 7(1):5–31, 2000.
16. A. Finkelstein, M. Harman, S. A. Mansouri, J. Ren, and Y. Zhang. "fairness analysis" in requirements assignments. *Requirements Engineering*, 14:231–245, 2008.
17. H. Geffner and J. Pearl. Conditional entailment: Bridging two approaches to default reasoning. *Artificial Intelligence*, 53(2-3):209–244, 1992.
18. D. Greer and G. Ruhe. Software release planning: an evolutionary and iteractive approach. *Information Software Technologies*, 46(4), 2004.
19. N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, C. Sierra, and M. Wooldridge. Automated negotiation: prospects, methods and challenges. *International Journal of Group Decision and negotiation*, 10(2):199–215, 2001.
20. U. Junker. Preference-based search for scheduling. In *Proceedings of AAAI'00*, pages 904–909, 2000.
21. U. Junker. Preference-based search and multi-criteria optimization. *Annals OR*, 130(1-4):75–115, 2004.
22. U. Junker. Preference-based search and multi-criteria optimization. *Annals of Operations Research*, 130:75–115, 2004.
23. U. Junker. Preference-based problem solving for constraint programming. In *Recent Advances in Constraints, 12th Annual ERCIM International Workshop on Constraint Solving and Constraint Logic Programming (CSCLP'07)*, pages 109–126, 2007.
24. C. O'Mahony and N. Wilson. Sorted-pareto dominance: an extension to the pareto dominance relation and its application in soft constraints. In *International Workshop on Preferences and Soft Constraints (SOFT'11)*, 2011.
25. A. Osyczka. Multicriteria optimization for engineering design. *Design Optimization*, pages 193–227, 1985.
26. K. Praditwong and X. Yao. A new multi-objective evolutionary optimization algorithm: the two-archive algorithm. In *Proceedings of the international conference on Computation Intelligent and Security (CIS'06)*, pages 286–291, 2006.
27. W. N. Robinson and V. Volkov. Requirements conflict restructuring. In *Georgia State University, Atlanta, GA*, 1999.
28. Y. Zhang, M. Harman, and S. A. Mansouri. The multi-objective next release problem. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'07)*, pages 1129–1136, 2007.