UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

Replacement planning tramite organizational mining e analisi di reti sociali

Replacement planning through organizational mining and social network analysis

Relatore:

Prof. Domenico Potena

Tesi Laurea di:

Marco Ciotti

Correlatore:

Dott. Emanuele Storti

ANNO ACCADEMICO 2018/2019

Indice

1. Introduzione

I sistemi informativi stanno diventando sempre più interconnessi con i processi operativi, riuscendo a registrare una grossa quantità di eventi durante l'esecuzione delle attività che supportano. Tuttavia, le organizzazioni hanno problemi ad estrarre valore da questi dati. L'obiettivo dell'organizational mining è quello di utilizzare i dati degli eventi per estrarre informazioni correlate ai processi, in particolare sulla struttura dell'organizzazione. Riuscire ad analizzare la rete sociale di un'organizzazione e descriverne le dinamiche può essere di notevole rilevanza per migliorare i processi produttivi e correggere eventuali errori nella struttura organizzativa. L'organizational mining, come vedremo in questa tesi, può avere un ruolo di fondamentale importanza anche nelle attività di gestione del personale. Quest'ultima infatti è una delle principali funzioni dell'azienda, qualsiasi siano le sue dimensioni. Tuttavia, quanto più aumenta la complessità dell'impresa, tanto maggiori sono le problematiche e le possibili conseguenze. Perciò la gestione del personale deve essere impeccabile in quanto è difatti fondamentale per incrementarne la produttività e l'efficienza.

I problemi più frequenti nella gestione del personale [1] sono:

- Turni scoperti ed eccessivi straordinari: gestire un elevato numero di dipendenti attraverso una pianificazione manuale, espone l'azienda a frequenti errori di pianificazione;
- Difficoltà ad avere una visione d'insieme dell'azienda: una pianificazione manuale difficilmente può essere impeccabile ed effettuare previsioni senza incorrere in errori;

- Difficoltà nell'elaborare il piano ferie;
- Troppi sprechi: un'inefficace gestione delle risorse, errori nella pianificazione ferie, dati mancanti e non aggiornati, esubero degli straordinari, comporta sprechi aziendali in termini economici e produttivi;
- Eccessiva conflittualità in azienda: una cattiva gestione del personale e della turnistica, genera caos e malcontento dei dipendenti, che andrà a compromettere la produttività aziendale.

Per la risoluzione di questi problemi l'organizational mining può essere di grande aiuto. Avere infatti molte più informazioni sulla rete sociale aziendale permette di prendere decisioni più efficienti, avendone una visione più chiara. Il mining sul processo non serve solamente ad avere uno strumento di supporto alle decisioni aziendali, ma permette inoltre la realizzazione di strumenti automatici per il controllo e per migliorare l'efficacia del processo.

In questo contesto, questo lavoro di tesi si pone l'obiettivo di fornire una soluzione valida per sostituire efficacemente ed efficientemente il personale. Considerando un'azienda in cui sono state programmate le attività che ogni persona deve eseguire, nel caso in cui una di queste si assentasse improvvisamente, è necessario avere un piano di riserva per mantenere il corretto funzionamento del processo aziendale.

Nello specifico si vuole:

1. Trovare la miglior soluzione possibile per sostituire una risorsa umana al momento non disponibile.

2. Identificare l'individuo migliore fra il personale in grado di ricoprire il ruolo della risorsa mancante.

Per il raggiungimento degli obiettivi si è seguito un approccio data-driven, cercando anche di considerare le relazioni sociali tra il personale dell'azienda. Per questo motivo sono state utilizzate le tecniche di organizational mining e analisi delle reti sociali. Lo svolgimento del lavoro ha attraversato le seguenti fasi. Per prima cosa, a partire dai dati sugli eventi è stato estratto il modello del processo. È stato poi prodotto il sociogramma dell'organizzazione, ovvero il grafo che rappresenta le relazioni sociali che intercorrono tra i membri dell'azienda., che è stato analizzato ed elaborato per estrarne le informazioni fondamentali. A questo punto, è stato specificato il problema e formulato un modello matematico necessario per la sua risoluzione. Il modello è stato risolto ed i risultati ottenuti sono stati analizzati e verificati per testarne la validità. Per eseguire tutto questo automaticamente e velocemente è stata sviluppata un'applicazione Java.

I vari capitoli di questa tesi illustrano e descrivono questo lavoro. Inizialmente, nel capitolo 2, vi è una breve descrizione e panoramica delle tecniche utilizzate e dei concetti di base. Nel capitolo 3 vi è una descrizione della metodologia e delle decisioni prese per arrivare alla soluzione. Nel capitolo 4 viene definito il caso di studio usato per implementare e testare l'applicativo software proposto. Viene spiegato nel capito 5 l'effettivo funzionamento dell'applicazione implementata durante questo lavoro. I risultati ottenuti in relazione al caso preso in esame e ad un modello di processo reale sono mostrati e analizzati nel capitolo 6. Infine, nel capitolo 7 vengono proposti gli eventuali sviluppi futuri e tratte le conclusioni.

2. Concetti fondamentali

Prima di illustrare il lavoro di tesi, è necessario descrivere le principali tecniche usate nel corso della sua realizzazione. Per tale motivo, in questo capitolo verrà fornita una panoramica sul Process e Organizzational Mining.

2.1 Process Mining

Il process mining è una disciplina relativamente giovane che permette l'analisi dei processi di business basata sui log degli eventi. Attraverso l'uso di specifici algoritmi di data mining applicati agli event log si può estrarre conoscenza da questi ultimi: è infatti possibile scoprire il modello e molte altre informazioni riguardanti un processo aziendale. L'obiettivo del process mining, infatti, è di migliorare quest'ultimo, fornendo tecniche e strumenti per la scoperta di strutture di processi, di dati, di organizzazioni e di strutture sociali a partire dai log [2].
L'analisi degli event log può essere utilizzata anche per confrontare i log con modelli a priori per studiare se quanto osservato sia conforme ad un modello descrittivo o prescrittivo.

Oggi c'è grande attenzione verso il BPM (Business process management), cioè l'insieme delle attività che ottimizzano, monitorano ed integrano i processi aziendali al fine di rendere efficace il business dell'azienda. Il BPM differisce dal BPR (Business Process Re-engineering), che toccò la sua massima diffusione negli anni novanta, perché mira ad un miglioramento incrementale dei processi, mentre il secondo ad un miglioramento radicale. Il process mining segue le fasi della progettazione dei processi aziendali

(Business Process Modeling), poi va oltre e fornisce anche un feedback per il suo miglioramento (BPM) [3]:

- l'**analisi dei processi** filtra, ordina e comprime i file di log per approfondire lo studio nel contesto delle operazioni dei processi;
- la **modellazione dei processi** può essere supportata dai feedback provenienti dal **monitoraggio dei processi** attraverso la registrazione di azioni o eventi (file di log);
- lo **sviluppo dei processi** sfrutta i risultati del process mining basati sui log per sviluppare ulteriori operazioni dei processi.

La figura 2.1 mostra che il process mining stabilisce i collegamenti tra gli attuali processi e i loro dati (event logs) e i modelli di processo (process model). L'universo digitale e l'universo fisico stanno diventando sempre più allineati. I sistemi di informazione di oggi registrano enormi quantità di eventi. Sistemi WFM classici (Workforce Management: ad esempio, Staffware e COSA), sistemi BPM (ad es. BPM | one di Pallas Athena, SmartBPM di Pegasystems, FileNet, Global 360 e Teamwork di Lombardi Software), sistemi ERP (Enterprise resource planning: ad esempio, SAP Business Suite, Oracle E-Business Suite e Microsoft Dynamics NAV), sistemi PDM (Product Data Management: ad esempio Wind-chill), sistemi CRM (Customer relationship management: ad es. Microsoft Dynamics CRM e SalesForce), middleware (ad esempio, IBM WebSphere e Cordys Business Operations Platform) e sistemi informativi ospedalieri (ad es. Chipsoft e Siemens Soarian) forniscono informazioni dettagliate sulle attività che vengono eseguite. La figura 2.1 si riferisce a tali dati come *event logs*. Tutti i sistemi appena citati forniscono direttamente tali event logs. Tuttavia, la maggior parte dei sistemi informativi

memorizza tali informazioni in forma non strutturata (ad esempio, i dati degli eventi sono sparsi su molte tabelle). In questi casi, esistono dati sugli eventi, ma sono necessari alcuni sforzi per estrarli. L'estrazione dei dati è parte integrante di qualsiasi attività di process mining.

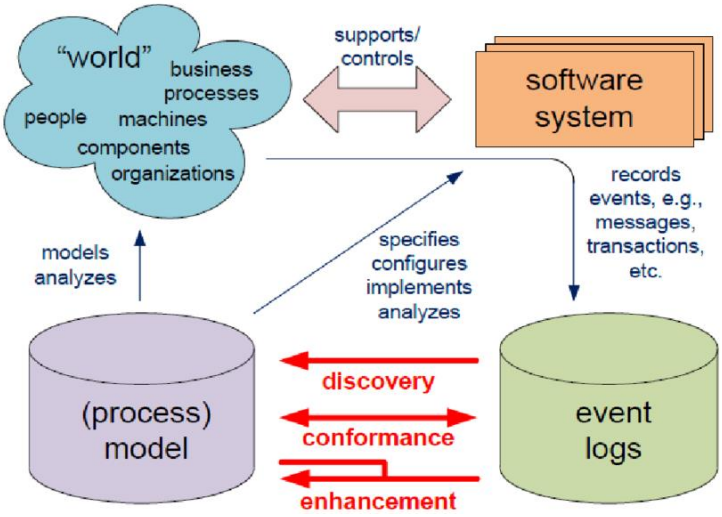


Fig. 2.1 Panoramica del process mining e delle principali attività: *discovery, conformance, and enhancement* [3]

La figura 2.1 inoltre ci mostra che ci sono tre tipi di attività di process mining.

La prima attività di process mining è la *discovery*. Una tecnica di discovery richiede un event log in input e produce in output un modello senza utilizzare alcuna informazione a priori. Il modello solitamente è sotto forma di rete di Petri: una rete di Petri (conosciuta anche come rete posto/transizione o rete P/T) è una delle varie rappresentazioni matematiche di un sistema distribuito discreto. Come un linguaggio di modellazione, esso descrive la struttura di un sistema distribuito come un grafo bipartito con delle annotazioni. Furono inventate nel 1962 durante la tesi di dottorato dell'autore Carl Adam Petri [4].

La seconda attività è la valutazione della *conformance*. Qui, un modello di processo esistente viene confrontato con un event log dello stesso processo. Il controllo della conformance può essere usato per verificare se la realtà, come registrato nel log, è conforme al modello e viceversa. Ad esempio, potrebbe esserci un process model che indica che l'acquisto di ordini superiori a un milione di euro richieda due assegni. L'analisi dell'event log mostrerà se questa regola è seguita o meno. Un altro esempio è il controllo del cosiddetto principio dei "quattro occhi" che afferma che determinate attività non possono essere svolte da una persona soltanto. Analizzando il registro degli eventi utilizzando un modello che specifica questi requisiti, si possono scoprire potenziali casi di frode.

Quindi, il conformance check può essere usato per rilevare, localizzare e spiegare deviazioni, misurandone la gravità.

La terza attività del process mining è l'*enhancement*. Qui, l'idea è di estendere o migliorare un process model esistente utilizzando le informazioni sul processo effettivo registrato in qualche event log. Mentre il controllo di conformità misura l'allineamento tra modello e realtà, questo terzo tipo di process mining mira a cambiare o estendere il modello. Un tipo di miglioramento è la *repair*, cioè la modifica del modello per riflettere meglio la realtà. Ad esempio, se due attività sono modellate in sequenza ma in realtà possono accadere in qualsiasi ordine, il modello può essere corretto. Un altro tipo di miglioramento è l'*extension*, cioè l'aggiunta di una nuova prospettiva al modello del processo confrontandolo con il log. Un esempio è l'estensione di un modello di processo con dati sulle prestazioni. Ad esempio, utilizzando i timestamp dell'event log si

possonono mostrare i colli di bottiglia, i livelli di servizio, i tempi di trasmissione e le frequenze.

2.2 Organizational Mining

L'organizational mining si riferisce a quelle attività di process mining volte a ricavare informazioni aggiuntive sull'organizzazione. Il mining organizzativo ha due obiettivi chiave [5]:

1. Sviluppare una comprensione dei social network che descrivono le interazioni tra le diverse risorse umane coinvolte nei processi;
2. Strutturare (o riorganizzare) i team in base ai ruoli e/o alle unità organizzative (IEEE Task Force on Process Mining 2011).

Analogamente al process mining, l'organizational mining si divide in *discovery*, *conformance* ed *enhancement*, sebbene con implementazioni differenti.

La fase di *discovery* si riferisce all'estrazione di un'organizational model o di un social network model che riflettono le impostazioni organizzative a partire dall'event log. Un organizational model di solito contiene informazioni riguanti le unità (ad es. i reparti), i ruoli del personale, gli originators dei processi e le relazioni (ad es. A è "part-of" con B ecc.). Il social network model invece descrive le interazioni fra le diverse risorse impiegate nell'esecuzione dei processi. Inoltre, questo tipo di mining può essere utilizzato in tandem con le altre informazioni correlate per scoprire le regole di assegnazione del lavoro, le regole di assegnazione delle risorse o le regole di profilazione degli utenti.

Il *conformance checking* confronta questi modelli / regole scoperti con i corrispondenti modelli / regole a priori.

L'*enhancement* nel contesto di mining organizzativo si riferisce all'arricchimento dei modelli / regole esistenti con informazioni aggiuntive, come ad esempio fornire process model astratti basati su diverse unità organizzative / ruoli.

L'organizational mining, comunque, fornisce una nuova dimensione per il process mining, che può fornire al team di gestione approfondimenti analitici sul contesto organizzativo. Ciò include informazioni basate sui dati riguardanti la struttura organizzativa e comunicazione, nonché una migliore comprensione degli indicatori delle prestazioni dei processi aziendali da un prospettiva della gestione delle operazioni organizzative. Tali informazioni sono difficili da estrarre con le attività tradizionali di process mining. I ricercatori nel dominio del process mining hanno riconosciuto l'importanza dell'organizational modeling e hanno proposto vari approcci.

Sellami et al. (2013) hanno proposto l'uso di un'ontologia organizzativa per annotare semanticamente gli event log per scoprire le relazioni tra gli esecutori delle attività. Il limite di questo approccio è quello di basarsi su una struttura di conoscenza sottostante statica, cioè l'ontologia organizzativa.

Song e van der Aalst (2008) propongono un approccio principalmente finalizzato al controllo della conformance dei modelli organizzativi: il loro l'approccio si basa sulle frequenze dei diversi mittenti che eseguono azioni simili per raccomandare il raggruppamento di tali originators nella stessa unità organizzativa.

In questo studio, viene adottato un approccio basato sull'estrazione del sociogramma che, identificando la struttura sociale dell'organizzazione, permette di derivare importanti informazioni sul personale e sulle relazioni che ci sono tra gli individui, correlate alle attività svolte.

3. Metodologia

Prima di descrivere le metodologia eseguita, è importante puntualizzare alcuni concetti fondamentali, dandone le precise definizioni per evitare eventuali ambiguità.

3.1 Definizioni

Per prima cosa è bene specificare la nozione di event log e traccia, seguendo la definizione data in [8].

3.1.1. Traccia ed event log

Sia A un insieme di attività e R un insieme di risorse. $V = A \times R$ è l'insieme di eventi possibili, cioè combinazioni di un'attività e una risorsa. Data una risorsa r, $V(r) \subseteq V$ è l'insieme di eventi in cui r può partecipare. Una traccia è una possibile sequenza di eventi, dove $C = V^*$ è l'insieme di tutte le tracce possibili. Un event log L è un sottoinsieme di tutte le tracce possibili, cioè $L = B(C)$, dove $B(C)$ è l'insieme di tutti gli insiemi (multi-set) oltre C.

Con il termine attività ci si riferisce a un'attività (o parte di essa) eseguita per raggiungere un obiettivo e con il termine risorsa a qualsiasi entità organizzativa che sia capace di svolgere alcune attività, incluso non solo il personale ma più in generale macchine, software, agenti.

Nel nostro scenario, il registro eventi contiene informazioni sull'attività eseguita da una certa risorsa e il tempo corrispondente. Da questa informazione è possibile derivare le capacità delle risorse e analizzare le relazioni tra le risorse come discusso nel seguito.

Di seguito una descrizione dettagliata delle attività e risorse, in termini di capacità di risorse, carico di risorse e richiesta media di carico per un'attività.

3.1.2. Capacità delle risorse

Data una risorsa $r \in R$, le capacità di r sono espresse come $A(r) \subseteq A$, cioè quelle attività che r è in grado di eseguire.

3.1.3. Richiesta di carico medio per attività

Dato un registro eventi L e un'attività $a \in A$, $\lambda(a) \in [0,1]$ è il suo carico medio richiesto ed è calcolata come $\lambda(a) = \frac{\sum_L execution\_time(a)}{\sum_L num\_occurrences(a)} \times \frac{1}{referenced\_period}$.

Qui, $execution\_time(a)$ è una funzione che restituisce il tempo di esecuzione per l'attività a in un'unità temporale e per un caso specifico, mentre la funzione $num\_occurrences(a)$ restituisce quante volte un'attività si verifica in una traccia. Infine, il rapporto è normalizzato dalla lunghezza del periodo di riferimento (ad es. 24 ore o 1 mese uomo).

3.1.4. Carico di lavoro di una risorsa

Le risorse possono essere caratterizzate in termini delle loro capacità (le attività che hanno svolto i passato) e il loro carico di lavoro. Quest'ultima è la misura del carico di lavoro corrente e massimo, come definito di seguito.

Data una risorsa $r \in R$, $\mu(r) \in [0,1]$ è il carico di lavoro massimo per r , mentre $\gamma(r) \in [0,1]$, con $\gamma(r) < \mu(r)$ è il fattore di carico di lavoro corrente di r.

In circostanze normali, per una risorsa r possiamo supporre che $\mu(r)$ sia uguale a 1, il che significa che possono essere assegnate attività per l'intero tempo di lavoro, ad es. 8 ore per un giorno lavorativo o 5 giorni per una settimana.

Per questa tesi si è supposto che sia conosciuto lo scheduling delle attività per ogni risorsa r nel periodo di riferimento di $\mu(r)$, cioè l'insieme delle attività che r deve svolgere durante il suo periodo di lavoro.

Lo scheduling viene definito come $S(r) \subseteq A^n$ dove n è il numero di attività da svolgere.

Quindi il carico di lavoro per la risorsa r è calcolabile come $\gamma(r) = \sum_{i=0}^{|S(r)|} \lambda(a_i)$

3.2. Sociogramma

Tra le varie relazioni che possono essere riconosciute in un event log tra due risorse, ci si concentra sulle relazioni di possibile causalità, e in particolare sulla consegna del lavoro (*handover of work*). All'interno di una traccia (ad esempio, l'istanza di processo) c'è un relazione in termini *handover of work* tra una risorsa r_1 e una risorsa r_2 se ci sono due attività successive a e b dove a è completata da r_1 e b da r_2. La metrica può essere definita e calcolata in più modi [8], in base al grado di causalità (diretto o indiretto), l'esistenza di una più successioni da una risorsa a se stessa (multiple self-transfer), e il tipo di successione. Il process model è necessario per riconoscere quale è il flusso corretto delle attività in una traccia.

In questo lavoro, ci si riferisce all'*handover of work* (1) ignorando gli auto-collegamenti, (2) considerando la successione indiretta e (3) la dipendenza casuale, vale a dire viene

presa in considerazione la successione tra le attività, con qualsiasi lunghezza, solo se allineata al process model. La ragione dietro (1) è che lo scopo del lavoro è legato alla sostituzione delle risorse, per il quale considerare l'handover of work della stessa risorsa non è utile. Il motivo di (2) e (3) è correlato al fatto che è necessario identificare correttamente il passaggio del lavoro nei processi reali ed evitare relazioni spurie.

Di seguito viene definita la funzione e la formula per il calcolo della metrica.

3.2.1. Handover of work

Dato a_1, a_2 ∈ A, r_1, r_2 ∈ R, un registro eventi L e n ∈ N, la funzione $r_1 \otimes^{\sigma}_{a1,a2} r_2$ restituisce quante volte r_1 e r_2 hanno eseguito rispettivamente l'attività a_1e a_2 dove la distanza tra a_1e a_2 è 1 nel modello del processo (nella rete di Petri non ci sono transizioni intermedie tra le transizioni a_1e a_2), per una determinata traccia σ ∈ L. Per l'intero registro degli eventi L, la funzione è calcolata come:

$$r_1 \otimes^{L}_{a_1,a_2} r_2 = \sum_{\sigma \in L} r_1 \otimes^{\sigma}_{a_1,a_2} r_2$$

Infine, la metrica dell'handover of work tra r_1 e r_2 per attività a_1e a_2 in L è calcolato come:

$$r_1 \odot^{L}_{a_1,a_2} r_2 = {r_1 \otimes^{L}_{a_1,a_2} r_2} \Big/ {\sum_{r_i \in R} \sum_{r_j \in R} r_i \otimes^{L}_{a1,a2} r_j}$$

La metrica è calcolata per una coppia di risorse r_1 , r_2 e rispetto una coppia di attività a_1, a_2 dividendo il numero totale di successioni casuali (senza auto-trasferimento) per il numero totale di successioni casuali di a_1e a_2 tra due risorse r_i , r_j con $i \neq j$.

L'handover of work non è stata una metrica determinante per la formulazione del modello matematico e per il calcolo dell'affinità tra le risorse. Piuttosto è stato un parametro che è servito nel calcolo dell'*handover matrix* (matrice che difatti rappresenta il sociogramma). Inoltre, risulta essere un parametro utile per l'analisi del sociogramma e lascia spazio ad ulteriori sviluppi futuri.

L'*handover matrix* è una matrice $N \times N$, dove $N < |R|$ è il numero di risorse attive (risorse cha hanno eseguito almeno 1 attività nel registro eventi).
Con due risorse $r_i, r_j \in R$ attive in L, la cella (i, j) della matrice di handover include una tupla (a_x, a_y, h_{ijxy}) se e solo se $h_{ijxy} = r_i \otimes^L_{a_x a_y} r_j > 0$.

3.2.2. Sociogramma

Data una matrice di handover M di dimensioni $N \times N$, un sociogramma G è definito come una tupla di 8 elementi $G = (\Sigma_{R'}, \Sigma_E, R', E, s, t, \ell_{R'}, \ell_E)$ dove

- $R' \subseteq R$ è l'insieme finito di N risorse nella matrice M ed E è un insieme di archi che rappresentano l'handover of work;
- $\Sigma_{R'}$e Σ_E alfabeti finiti delle etichette dei vertici e degli archi disponibili, $s: E \rightarrow R'$ e $t: E \rightarrow R'$ sono due funzioni che mappano il nodo iniziale e finale di un arco alle corrispondenti risorse;

- $\ell_{R'}: R' \rightarrow \Sigma_{R'}$ e $\ell_E: E \rightarrow \Sigma_E$ sono due funzioni che mappano i vertici e archi con la rispettiva etichetta.

Nel seguito, per semplicità, un sociogramma G viene trattato come una tupla $G = (R', E)$, con E multiset di archi. Ogni arco $e \in E$ sarà brevemente rappresentato come una tupla $e = \left(n_i, n_j, \left(a_x, a_y, h_{ijxy}\right)\right)$ che collega due nodi n_i, $n_j \in R'$ e $\left(a_x, a_y, h_{ijxy}\right)$ è l'etichetta dell'arco.

Nella figura 3.1 viene mostrato un esempio di sociogramma.

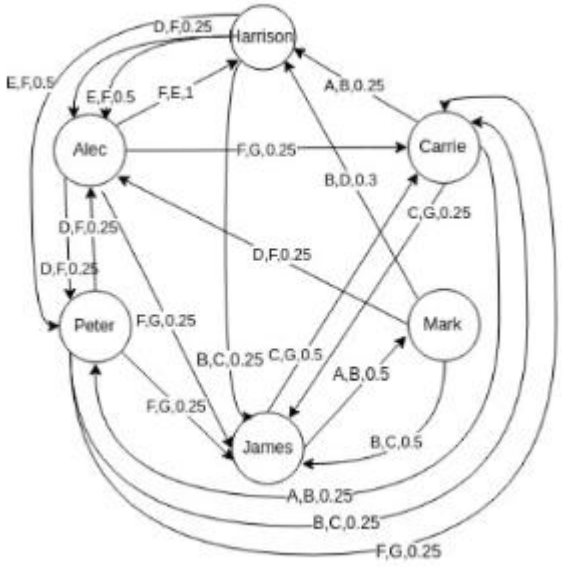


Fig. 3.1 Esempio di un sociogramma semplice

Si prega di notare che la definizione di sociogramma che data è diversa da quella tipicamente indicata nella letteratura sull'organizational mining. Anzi, ci si riferisce qui ad un multigrafo etichettato con più lati tra due nodi. Questo consente una maggiore

espressività in quanto si può codificare non solo l'handover generico tra due risorse, ma anche tenere conto di quali compiti specifici una risorsa ha consegnato all'altra.

3.3. Modello matematico

Una volta ottenuto il sociogramma, la fase di sostituzione delle risorse volge a identificare effettivamente la/e risorsa/e da sostituire con quella mancante.
Di seguito si definisce il problema e un modello per la sua soluzione.

3.3.1. Definizione del problema

Dato un sociogramma G, una risorsa non disponibile r da sostituire e un insieme $\{a_1, \ldots, a_m\}$ di attività da assegnare; dato, per ogni risorsa $r_i \in R$ il suo fattore di carico di lavoro $\lambda(r_i)$ (calcolato come definito sopra), il fattore di carico massimo $\mu(r_i)$, le sue capacità $A(r_i)$, il problema è definito come segue:

- Determinare un insieme di risorse $\{r_1, \ldots, r_n\}$, con $r_i \in R$ che è collettivamente capace di sostituire r per eseguire $\{a_1, \ldots, a_m\}$ sotto una serie di vincoli.

Per prima cosa identifichiamo i seguenti requisiti e vincoli per una soluzione:

- affinità: le risorse più compatibili con quella da sostituire sono preferibili.
- disponibilità: una risorsa può essere selezionata solo se il suo fattore di carico di lavoro è sufficiente per eseguire l'attività richiesta.
- minimalità: si preferiscono soluzioni che prevedono un insieme minimo di risorse;

- bilanciamento del carico: le risorse con un basso carico di lavoro sono preferibili rispetto a quelle con un alto fattore, per migliorare il bilanciamento del carico delle risorse.

3.3.2. Formulazione del modello matematico

Dato un sociogramma G e una risorsa $r \in R$ da sostituire, sia $S(r)$ il multiset[1] di attività assegnate a r che deve essere eseguito da altre risorse[2]. I seguenti passaggi descrivono come viene definito il modello:

1. Determinare l'insieme RC delle risorse candidate con almeno una capacità uguale a un'attività in $S(r)$: $RC = \{r_i \in R : \exists\, a_j \in A\,(r_i)\ tale\ che\ a_j \in S(r)\}$
2. $\forall\, r_i \in RC,\ \forall\, a_j \in S(r)$, x_i^j è una variabile binaria definita come segue:
    - 0 non è selezionato per la sostituzione di r per eseguire attività a_j
    - 1 viene selezionato per la sostituzione di r per eseguire attività a_j
3. Viene definito il seguente modello di programmazione lineare intera (ILP) 0-1, assumendo che $|\, RC \,| = q$ e $|\, S(r) \,| = s$:

$$(1) \quad min \sum_{i=1}^{q} l_i \cdot \left(\sum_{j=1}^{s} c_i^j \cdot x_i^j \right)$$

$$(2) \quad \boldsymbol{A^T \cdot X = 1}$$

[1] S(r) è un multiset perché potrebbe essere stata assegnata a r una stessa attività più volte in un processo.

[2] Poiché l'algoritmo di replacement può essere rieseguito in caso di necessità, non vi è alcun vincolo sulla dimensione di questo set.

$$(3) \quad \boldsymbol{X} \cdot \boldsymbol{\delta} + \boldsymbol{\gamma} \leq \boldsymbol{\mu}$$

$$(4) \quad x_i^j \in \{0, 1\}$$

dove $l_i \in [0, 1]$ è il fattore di costo che tiene conto del bilanciamento del carico, mentre $c_i^j \in [0, 1]$ è un fattore di costo che tiene conto dell'affinità tra la risorsa candidata per sostituire r_i e per l'esecuzione della specifica attività a_j. Essi saranno entrambi discussi più avanti in questa sottosezione. L'espressione (1) è la funzione obiettivo da minimizzare al fine di soddisfare i requisiti di affinità, bilanciamento del carico e minimalità: il minor numero di risorse compatibili sarà selezionato, ponderato in base ai fattori di costo. L'espressione (2) è un sistema di k disuguaglianze con s variabili ciascuna e ha lo scopo di verificare che ogni attività selezionata dell'insieme T venga eseguita almeno da una risorsa disponibile. La matrice A è tale che $A\,[i]\,[j] = 1$ se $a_j \in A(r_i)$, 0 altrimenti. Infine, espressione (3) è un sistema di disuguaglianze volto a soddisfare i requisiti di disponibilità. Infatti, ogni disuguaglianza rappresenta il vincolo in base al quale alcune attività possano essere eseguite da una risorsa nella sua capacità rimanente. Più formalmente, limita l'assegnazione di alcune attività a una risorsa disponibile r_i solo se la somma del carico di lavoro medio totale di tali attività, meno il carico di lavoro corrente $\gamma(r_i)$ è inferiore al carico di lavoro massimo $\mu(r_i)$ per esso. Infine, l'espressione (4) lega le variabili ad assumere un valore binario (0 o 1), cioè che rispettivamente il compito non è stato assegnato o è stato assegnato a quella risorsa.

3.3.3. Fattori di costo

I fattori di costo l e c utilizzati nel modello sono rispettivamente mirati per pesare ciascuna risorsa candidata in termini di (i) carico di lavoro e (ii) affinità con la risorsa da sostituire. Come spiegato in seguito, mentre il carico di lavoro di r_i è considerato per preferire risorse a basso carico di lavoro (requisito di bilanciamento), l'affinità è misurata attraverso una metrica di similitudine comparando r con r_i nel il sociogramma (requisito di affinità). Questo approccio è globalmente mirato a selezionare risorse simili a quella da sostituire, in termini delle capacità, dell'esperienza e della velocità di esecuzione del lavoro, ma allo stesso tempo bilanciare il carico di lavoro tra tutte le risorse. Entrambi i fattori $0 \leq l$ e $c \leq 1$ sono tali da ridurre il costo, aumentando le probabilità che la risorsa venga selezionata per la sostituzione; difatti la funzione obiettivo è una funzione di minimo.

Data una risorsa per sostituire r e una risorsa candidata r_i, il fattore di costo l è equivalente al carico di lavoro rimanente per la risorsa r_i, ovvero:

$$l_i = 1 - \beta \cdot \big(\mu(r_i) - \gamma(r_i)\big)$$

β è un coefficienti usato per pesare questo fattore di costo.

In particolare, l'intervallo va da 0 (quando $\gamma(r_i) = \mu(r_i)$ la disponibilità delle risorse è esaurita), a $\mu(r_i)$ (nel caso in cui la risorsa sia completamente disponibile).

D'altra parte, data un'attività a_j, il costo fattore c_i^j il è definito come segue:

$$c_i^j = 1 - sim_j(r, r_i)$$

3.3.4. Similarità

La funzione sim restituisce il grado di affinità tra il due risorse per l'esecuzione dell'attività a_j. L'espressione per il calcolo della funzione è la seguente:

$$sim_j(r_h, r_k) = v_j(r_h, r_k) \cdot \frac{w_1 \cdot collaboration_j(r_h, r_k) + w_2 \cdot speed_j(r_h, r_k) + w_3 \cdot experience_j(r_h, r_k)}{w_1 + w_2 + w_3}$$

Dove $v_j(r_h, r_k) \in \{0,1\}$ ed è uguale a 0 se almeno una delle due risorse non ha mai eseguito nel log l'attività a_j, 1 altrimenti. I pesi $\{w_1, w_2, w_3\}$ sono associati rispettivamente ai tre fattori di similarità.

Le tre funzioni hanno codominio $[0,1]$ quindi anche la similarità ha lo stesso codominio e un valore massimo pari a 1.

La funzione $collaboration$ indica il grado di affinità tra r_h e r_k riguardo le collaborazioni con le altre risorse sull'attività a_j.

Dato R' con cardinalità n, insieme delle risorse con cui ha collaborato r_h, cioè collegate tramite il sociogramma da almeno un lato uscente da r_h con etichetta $(a_j, .., ..)$).

Dato R'' insieme di risorse con cui ha collaborato r_k, e l'intersezione tra R' e R'' di cardinalità m, abbiamo che $m < n$. La funzione è la seguente:

$$collaboration_j(r_h, r_k) = \frac{m}{n}$$

La funzione $speed$ mette in relazione la velocità di esecuzione dell'attività a_j da parte delle due risorse. Dati t_h come tempo medio di esecuzione dell'attività a_j da parte di r_h e t_k tempo medio di esecuzione da parte di r_k, la funzione è definita come segue:

$$speed_j(r_h, r_k) = \begin{cases} 1, & t_h \geq t_k \\ \frac{t_h}{t_k}, & t_h < t_k \end{cases}$$

La funzione $experience$ confronta invece il livello di esperienza delle due risorse riguardo l'attività a_j. Dato q_h come numero di volte in cui r_h ha eseguito la suddetta attività, ed q_k il numero di volte in cui l'ha eseguita r_k. La definizione è la seguente:

$$experience_j(r_h, r_k) = \begin{cases} 1, & q_k \geq q_h \\ \frac{q_k}{q_h}, & q_k < q_h \end{cases}$$

3.3.5. Affinità tra due risorse

La formulazione del modello permette di risolvere il problema legato alla sostituzione di alcune delle attività della risorsa mancante. Questo è molto utile nel caso in cui la risorsa non sia disponibile temporaneamente e le attività erano già state programmate. Se però si deve sostituire permanentemente la risorsa, occorre avere una misura di affinità non solo legata alle singole attività, ma in maniera più generale tra le risorse.

L'affinità è stata quindi così definita:

$$affinity(r_h, r_k) = \frac{w_0 \cdot activity(r_h, r_k) + w_1 \cdot collaboration(r_h, r_k) + w_2 \cdot speed(r_h, r_k) + w_3 \cdot experience(r_h, r_k)}{w_0 + w_1 + w_2 + w_3}$$

Anche in questo caso w_0, w_1, w_2, w_3 rappresentano i pesi legati ai singoli fattori, ma va specificato che, per l'efficacia del calcolo, w_0 e w_1 devono essere almeno di un ordine di grandezza superiore rispetto agli altri due.

Date le capacita delle due risorse $A(r_h)$ e $A(r_k)$, p come la cardinalità di $A(r_h) \cap A(r_k)$ ed q come la cardinalita di $A(r_h)$.

$$activity(r_h, r_k) = \begin{cases} 0, & q = 0 \\ \frac{p}{q}, & q \neq 0 \end{cases}$$

La definizione di $collaboration$ è uguale al paragrafo 4.3.4, con la differenza che si considerano le risorse collegate da almeno un lato del sociogramma, non è importante la sua etichetta.

Dato $A(r_h) \cap A(r_k)$ insieme di attività in comune fra r_h ed r_k di cardinalità p, si definiscono:

$$speed(r_h, r_k) = \frac{\sum_{j \in A(r_h) \cap A(r_k)} e_j}{p} \qquad \text{con} \qquad e_j = \begin{cases} 1, & t_j(r_k) < t_j(r_h) \\ 0, & altrimenti \end{cases}$$

dove $t_j(r_h)$ rappresenta il tempo medio di esecuzione da parte di r_h per l'attività a_j;

$$experience(r_h, r_k) = \frac{\sum_{j \in A(r_h) \cap A(r_k)} f_j}{p} \qquad \text{con} \qquad f_j = \begin{cases} 1, & n_j(r_h) < n_j(r_k) \\ 0, & altrimenti \end{cases}$$

dove $n_j(r_h)$ rappresenta il numero di volte in cui è stata eseguita l'attività a_j da parte di r_h.

4. Caso di studio

Come già detto, il fine principale di questa tesi è quello di fornire un supporto per la sostituzione del personale in un azienda.

Per le attività di organizational mining e per lo sviluppo dell'applicazione è stato considerato inizialmente un caso di studio semplice, con dati fittizi, per poter testare e valutare più velocemente i risultati ottenuti. Una volta verificata la validità della soluzione, si è poi passati ad analizzare e applicare quest'ultima su di un processo reale.

4.1 Descrizione

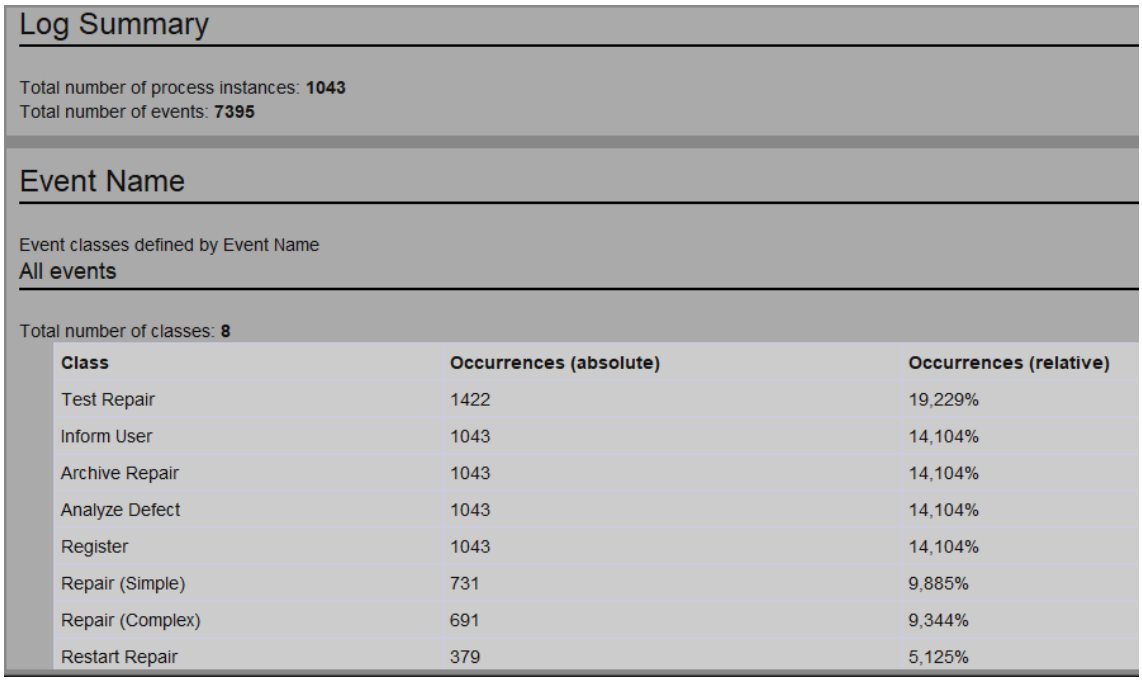
Il caso di studio preso in considerazione riguarda un'azienda che si occupa della ripazione di prodotti. L'event log di questo processo è reperito sul sito di ProM [6] nella sezione "Example log files".

Le attività che vengono svolte nell'esecuzione del processo sono di 8 tipi:

1. Register: all'inizio di ogni processo viene effettuata la registrazione del prodotto e l'avvio della pratica di riparazione;
2. Inform user: l'utente proprietario del prodotto vieni informato sullo stato del processo;
3. Analyze Defect: viene analizzato i prodotto per capirne i difetti ed il motivo del suo malfuzionamento;
4. Repair (simple): il prodotto non presenta gravi problemi quindi viene effettuata una riparazione "semplice" che può essere affidata anche ad un personale meno esperto;

5. Repair (complex): il prodotto presenta problemi specifici e complessi e la pratica viene affidata ad una risorsa più esperta;
6. Test repair: completata la riparazione il prodotto viene opportunatamente testato;
7. Restart repair: se il test viene fallito deve essere riavviata la procedura di riparazione;
8. Archive repair: se il prodotto funziona correttamente la pratica viene archiviata.

La figura 4.1 riporta le informazioni dell'event log del caso di studio.



Log Summary

Total number of process instances: **1043**
Total number of events: **7395**

Event Name

Event classes defined by Event Name
All events

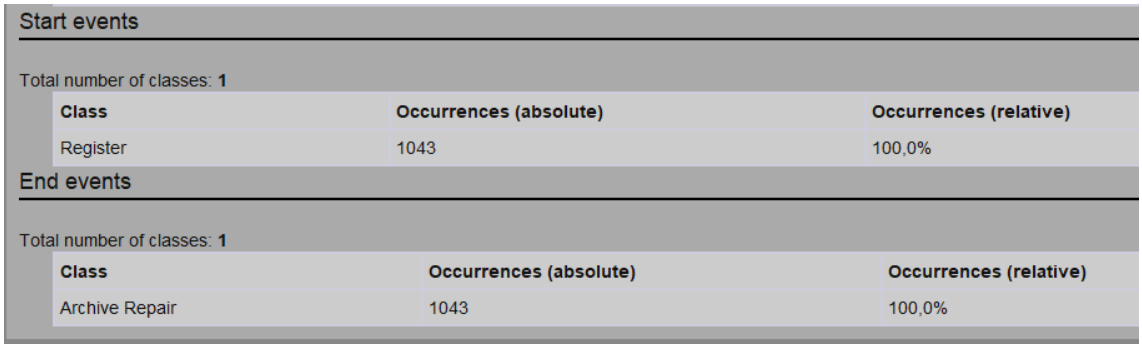
Total number of classes: **8**

Class	Occurrences (absolute)	Occurrences (relative)
Test Repair	1422	19,229%
Inform User	1043	14,104%
Archive Repair	1043	14,104%
Analyze Defect	1043	14,104%
Register	1043	14,104%
Repair (Simple)	731	9,885%
Repair (Complex)	691	9,344%
Restart Repair	379	5,125%

Fig. 4.1 Informazioni relative all'event log del processo di ripazione.

Come è possibile vedere dalla figura 4.1, l'event log ha una dimensione relativamente piccola con un numero totale di eventi pari a 7395 e 1043 tracce totali. La figura mostra inoltre i vari tipi di attività del processo con le relative occorrenze (assolute e in percentuale).

Nella figura 4.2 invece è possile constatare che il cento percento delle tracce inizia con un attività di Register e termina con Archive Repair.



Start events

Total number of classes: **1**

Class	Occurrences (absolute)	Occurrences (relative)
Register	1043	100,0%

End events

Total number of classes: **1**

Class	Occurrences (absolute)	Occurrences (relative)
Archive Repair	1043	100,0%

Fig. 3.2 Eventi di inizio e fine.

4.2 Data set

Il data set di partenza è stato ricavato da un file .csv. Dalla figura 4.3 è possibile vedere come ogni riga del file excel rappresenta un evento.

case	event	startTime	completeTime
1	Register	1970/01/02 12:23:00.000	1970/01/02 12:23:00.000
1	Analyze Defect	1970/01/02 12:30:00.000	1970/01/02 12:30:00.000
1	Repair (Complex	1970/01/02 12:31:00.000	1970/01/02 12:31:00.000
1	Test Repair	1970/01/02 12:55:00.000	1970/01/02 12:55:00.000
1	Inform User	1970/01/02 13:10:00.000	1970/01/02 13:10:00.000
1	Archive Repair	1970/01/02 13:10:00.000	1970/01/02 13:10:00.000
2	Register	1970/01/01 21:47:00.269	1970/01/01 21:47:00.269
2	Analyze Defect	1970/01/01 21:48:15.089	1970/01/01 21:48:15.089
2	Inform User	1970/01/01 21:49:29.985	1970/01/01 21:49:29.985
2	Repair (Complex	1970/01/01 21:50:44.955	1970/01/01 21:50:44.955
2	Test Repair	1970/01/01 21:52:00.000	1970/01/01 21:52:00.000
2	Archive Repair	1970/01/01 21:57:00.000	1970/01/01 21:57:00.000

Fig. 4.3 File excel con l'insieme dei dati

La colonna case identifica il numero della traccia, l'event il tipo di attività svolta, lo start ed il complete time rappresentano il tempo di inizio e fine attività.

4.3 Attività di pre-processing

L'insieme dei dati è stato ottenuto da un file .csv da cui sono state eseguite delle attività di pre-processing per ricaverne l'event log e il modello.

Per lo sviluppo del lavoro sono state fatte due assunzioni:

1. L'event log deve essere conforme al modello del processo;
2. Gli eventi devono avere informazioni sulla risorsa che lo ha eseguito e la data in cui è stato generato.

Per quanto riguarda il punto 2, inizialmente mancavano le informazioni sulle risorse. È stata generata artificialmente una colonna "resource" contenente il nome della risorsa che ha compiuto quell'attività. I nomi delle risorse sono stati parametrizzati con il carattere "R" seguito da un numero. Ai fini del progetto non è stata importante la validità di questi dati in quanto, come già detto, il caso di studio è servito per lo sviluppo e il testing dell'applicazione. Perciò l'informazione sulle risorse è stata aggiunta definendo due team all'interno dell'azienda:

- Il primo team composto dalle risorse R1 ed R2 si occupa dell'esecuzione di attività di amministrazione, ovvero Register, Inform User e Archive Repair.
- Il secondo composto da R3, R4, R5, R6 esegue le attività che richiedono competenze più tecniche, ovvero Analyze Defect, Repair (Simple e Complex), Test Repair e Restart Repair;
- All'interno del secondo team si è pensata un'ulteriore suddivisione: le attività che richiedono maggiore competenza, cioè Analyze Defect e Restart Repair, sono eseguibili soltanto da R3 ed R4.

Secondo quanto detto è stata definita la probabilità associata ad ogni risorsa per eseguire una determita attività. Nella figura 4.4, "rx" è la colonna che è servita per definire le

probabilità attraverso la formula seguente (valida per la riga 2, per le altra cambia il riferimento alla colonna B):

=IF(OR(B2="Register";B2="InformUser";B2="ArchiveRepair");IF(RAND()>0,5;"R1";"R2");
IF(OR(B2="AnalyzeDefect";B2="RestartRepair");IF(RAND()>0,5;"R3";"R4");"))

Tramite questa formula e l'ausilio della colonna "casuale" che contiene un numero casuale che va da 0 a 1, è stato possibile associare casualmente le risorse alle attività, rispettando comunque le regole sopra definite. La figura 4.4 seguente mostra i cambiamenti.

	A	B	C	D	E	F	G
1	case	event	startTime	completeTime	rx	casuale	resource
2	1	Register	1970/01/02 12:23:00.000	1970/01/02 12:23:00.000	R2	0,543352	R2
3	1	Analyze Defect	1970/01/02 12:30:00.000	1970/01/02 12:30:00.000	R3	0,88036	R3
4	1	Repair (Complex	1970/01/02 12:31:00.000	1970/01/02 12:31:00.000		0,234497	R3
5	1	Test Repair	1970/01/02 12:55:00.000	1970/01/02 12:55:00.000		0,843831	R6
6	1	Inform User	1970/01/02 13:10:00.000	1970/01/02 13:10:00.000	R1	0,91125	R1
7	1	Archive Repair	1970/01/02 13:10:00.000	1970/01/02 13:10:00.000	R1	0,203753	R1
8	2	Register	1970/01/01 21:47:00.269	1970/01/01 21:47:00.269	R1	0,908231	R1
9	2	Analyze Defect	1970/01/01 21:48:15.089	1970/01/01 21:48:15.089	R4	0,469263	R4
10	2	Inform User	1970/01/01 21:49:29.985	1970/01/01 21:49:29.985	R2	0,737107	R2
11	2	Repair (Complex	1970/01/01 21:50:44.955	1970/01/01 21:50:44.955		0,227517	R3
12	2	Test Repair	1970/01/01 21:52:00.000	1970/01/01 21:52:00.000		0,367921	R4
13	2	Archive Repair	1970/01/01 21:57:00.000	1970/01/01 21:57:00.000	R2	0,494073	R2
14	4	Register	1970/01/02 00:46:00.000	1970/01/02 00:46:00.000	R1	0,600401	R1
15	4	Analyze Defect	1970/01/02 00:53:00.000	1970/01/02 00:53:00.000	R4	0,640068	R4
16	4	Repair (Simple)	1970/01/02 01:21:00.000	1970/01/02 01:21:00.000		0,993579	R6
17	4	Inform User	1970/01/02 01:29:00.000	1970/01/02 01:29:00.000	R2	0,462756	R2
18	4	Test Repair	1970/01/02 01:42:00.000	1970/01/02 01:42:00.000		0,919128	R6
19	4	Archive Repair	1970/01/02 01:50:00.000	1970/01/02 01:50:00.000	R1	0,839512	R1
20	5	Register	1970/01/02 02:41:00.000	1970/01/02 02:41:00.000	R2	0,51547	R2
21	5	Analyze Defect	1970/01/02 02:49:00.000	1970/01/02 02:49:00.000	R3	0,063414	R3
22	5	Repair (Complex	1970/01/02 02:53:00.000	1970/01/02 02:53:00.000		0,898929	R6
23	5	Test Repair	1970/01/02 03:15:00.000	1970/01/02 03:15:00.000		0,667741	R5

Fig. 4.4 Data-set con l'insieme delle risorse

Riguardo alle informazioni sulla data e l'orario, erano già presenti nel file, però presentavano delle incongruenze: attività svolte dopo di altre avevano uno startTime minore (cioè erano iniziate prima). Per questo si è deciso di riordinare i tempi aggiungendo due colonne come mostrato dalla figura 4.5.

	A	B	C	D	E	F	G	H	I	J
1	case	event	startTime	completeTime	rx	casuale	resource	startTimeOrdinato	completeTimeOrdinato	
2	1	Register	1970/01/02 12:23:00.000	1970/01/02 12:23:00.000	R1	0,80123	R1	1970/01/01 21:16:33.939	1970/01/01 21:16:33.939	
3	1	Analyze Defect	1970/01/02 12:30:00.000	1970/01/02 12:30:00.000	R3	0,791316	R3	1970/01/01 21:17:46.933	1970/01/01 21:17:46.933	
4	1	Repair (Complex	1970/01/02 12:31:00.000	1970/01/02 12:31:00.000		0,131528	R3	1970/01/01 21:19:00.000	1970/01/01 21:19:00.000	
5	1	Test Repair	1970/01/02 12:55:00.000	1970/01/02 12:55:00.000		0,142456	R3	1970/01/01 21:20:33.459	1970/01/01 21:20:33.459	
6	1	Inform User	1970/01/02 13:10:00.000	1970/01/02 13:10:00.000	R1	0,552401	R1	1970/01/01 21:21:46.693	1970/01/01 21:21:46.693	
7	1	Archive Repair	1970/01/02 13:10:00.000	1970/01/02 13:10:00.000	R1	0,23686	R1	1970/01/01 21:23:00.000	1970/01/01 21:23:00.000	

Fig. 4.5 Aggiunta di starTimeOrdinato e completeTimeOrdinato

In questo modo, si sono risolti il problemi di incongruenza ed è stato possibile estrarre dal .csv un file .xes (formato standard per gli event log) che presentasse l'orario e la data relativi all'evento.

È facilmente intuibile come le informazioni sulle risorse siano state necessarie per la realizzazione del lavoro. Gli orari e le date sono serviti per estrarre i tempi di esecuzione delle attività. Inoltre, è stata fatta un'ulteriore assunzione: poiché lo start ed il complete time erano identici, è stato considerato per la creazione dello .xes soltanto lo startTime. Nei successivi capitoli verrà illustrato come sono stati ricavati i tempi di esecuzione.

Il file .xes è stato generato tramite ProM [6]. ProM è un framework estendibile che supporta un'ampia varietà di tecniche di process mining sotto forma di plug-in. È indipendente dalla piattaforma poiché è implementato in Java e può essere scaricato gratuitamente. Nello specifico è stato eseguito il plug-in "Convert CSV to XES" (di F. Mannhardt).

4.4 Modello del processo

Il modello del processo, sotto forma di rete di Petri, è stato ricavato utilizzando l'algoritmo Alpha Miner. L' algoritmo α è un algoritmo utilizzato nel process mining, volto a ricostruire la causalità da un insieme di sequenze di eventi . Fu proposto per la prima volta da van der Aalst , Weijters e Măruşter [7].

La reti di Petri ottenuta è mostrata nella figura 4.6.

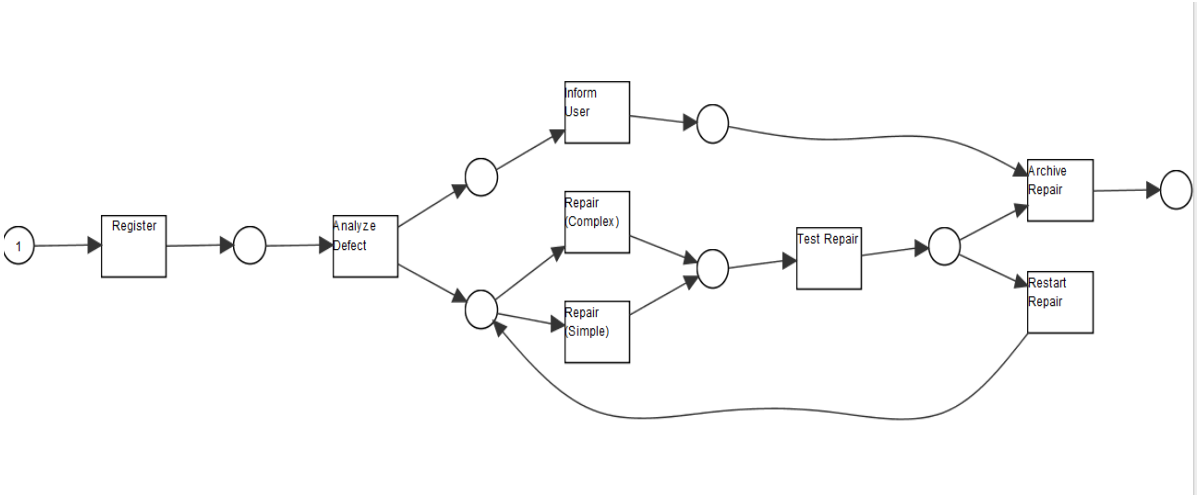


Fig. 4.6 Modello del processo di ripazione

Come si può constatare, il modello del processo è piuttosto semplice e facilmente comprensibile. Ciò ha agevolato notevolmente la sviluppo del lavoro.

L'event log, inoltre, risulta essere perfettamente conforme al modello come evidenziato dai risultati di conformance checking riportati nella figura 4.7.

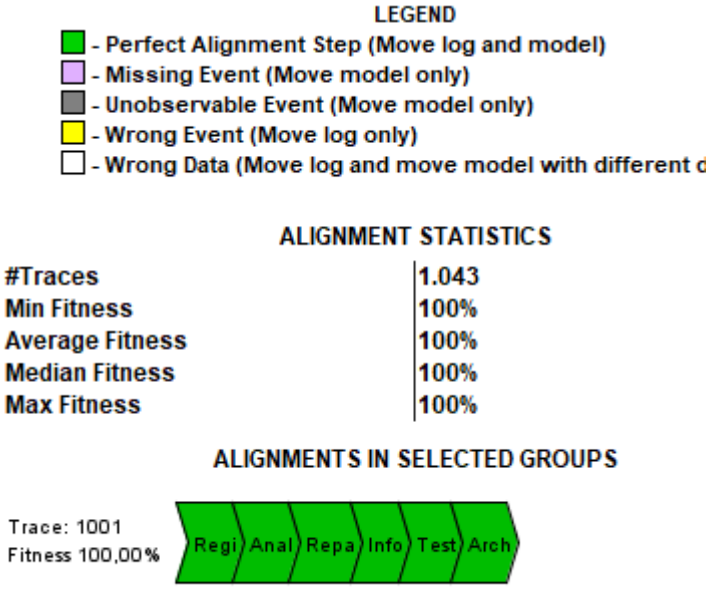


Fig. 4.7 Risultati del conformance checking

La figura mostra come tutte le tracce hanno il 100% di fitness con il modello.

A questo punto, avendo ottenuto gli input, cioè l'event log e la rete di Petri che rispettano le assunzioni descritte in 4.3, è stato possibile iniziare lo sviluppo dell'applicazione.

5. Applicazione software

5.1. Descrizione

L'applicazione software prodotta durante lo svolgimento di questo lavoro di tesi, è uno strumento di supporto per il replacement planning. L'applicazione è stata sviluppata in Java [9] e si relaziona con un database MySQL [10]. Essa prende in input l'event log e il modello del processo (che devovo soddisfare i due requisiti specificati nel paragrafo 4.3), la risorsa che si vuole sostituire, le attività che dovevano essere svolte e restituisce in output un file excel contenente la soluzione del modello matematico (3.3) e le risorse ordinate per percentuale di affinità con la risorsa mancante (3.3.5). L'applicazione si compone essenzialmente di due parti: la prima estrae dall'event log e dal modello il sociogramma (3.2) sia sotto forma di grafo che come matrice di handover (3.2.1); la seconda parte ricava dal sociogramma le informazioni necessarie per la risoluzione del modello matematico, risolve il modello e restituisce i dati. Il funzionamento sarà spiegato in dettaglio nel paragrafo successivo. Durante la fase di sviluppo dell'applicazione non è stata costruita un'interfaccia grafica per l'utente. L'obiettivo del progetto era infatti quello di fornire un'applicazione di base funzionante; per questo la progettazione e la creazione dell'UI sono state lasciate ad eventuali sviluppi futuri.

5.2. Funzionamento

5.2.1. Estrazione del sociogramma

La prima parte dell'applicazione estrae il sociogramma dagli event log e dal modello. Per ottenere questo, effettuando il replay del log sul modello, vengono costruiti gli instance graphs [11], cioè dei grafi che mostrano esplicitamente i parallelismi che sono nascosti nelle traccie sequenziali. Nella figura 5.1 vi è riportato un esempio di instance graph relativo al caso di studio.

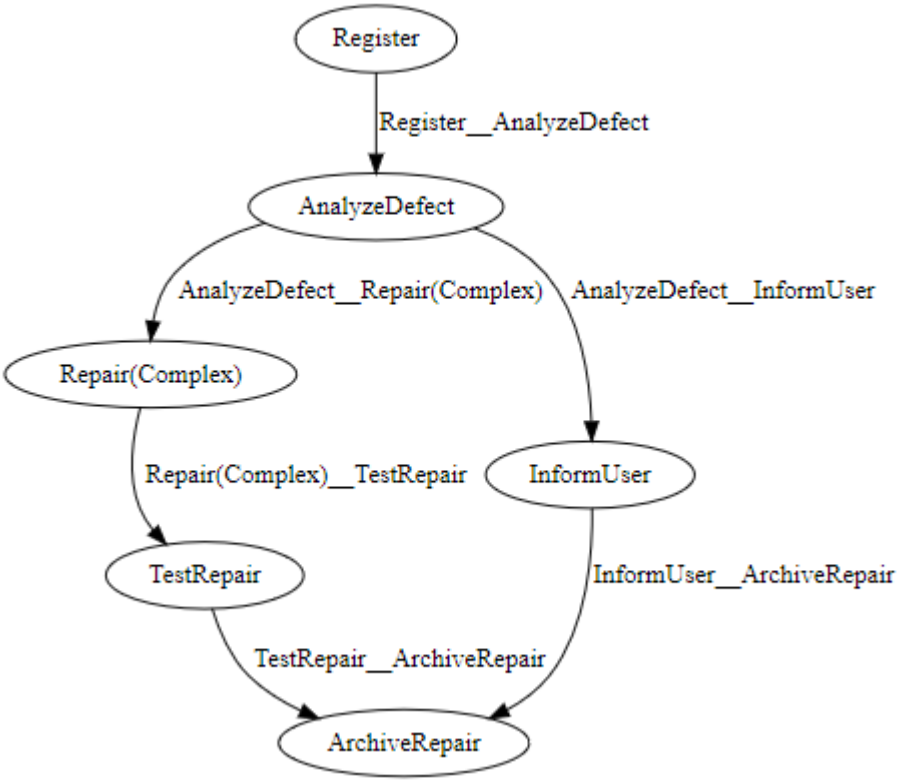


Fig. 5.1 Esempio di instance graph relativo a (4).

Come mostrato nella figura, ad es. le attività Inform User e Test Repair sono chiaramente attività eseguite in parallelo, anche se nelle traccie dell'event log compaiono come eventi

sequenziali. Per questo motivo si fa riferimento agli instance graph per la costruzione del sociogramma, altrimenti le relazioni fra gli utenti risulterebbero essere non veritiere. Ad esempio, nel registro potrebbero comparire in sequenza l'attività Inform User svolta da R1 e l'attività di Test Repair svolta da R4. Basandosi soltanto sull'event log, si potrebbe affermare che le due risorse abbiano una relazione per queste attività. Invece, grazie all'instance graph della fig. 5.1 si nota palesemente come le due risorse non abbiano collaborato nella consegna del lavoro, mentre ad es. chi ha compiuto l'ArchiveRepair si è relazionato sia con R1 e sia con R4.

Per l'estrazione di questi grafi è stato usato il codice B.I.G. (Building Instance Graphs) sviluppato da Laura Genga[3], adattandolo al contesto e rimuovendo la parte di conformance checking.

Dagli instance graphs sono state estratte le informazioni necessarie alla costruzione della matrice del sociogramma, memorizzata nel database con la seguente tabella riportata in figura 5.2. Cioè per ogni res_1 e per ogni res_2 appartenti all'insieme R delle risorse, ci si è calcolato l'*handover of work* su ogni attività $activity_1 \in A(res_1)$ e ogni attività $activity_2 \in A(res_2)$, con informazioni aggiuntive sul numero di occorrenze e sul tempo.

[3] Laura Genga è una ricercatrice del Department of Industrial Engineering & Innovation Sciences, Information Systems IE&IS del TU/E Eindhoven University of Technology. Collabora col Dipartimento di Ingegneria dell'Informazione dell'Università Politecnica delle Marche, dove si è laureata in informatica e automazione ingegneria nel 2012.

id	res1	res2	activity1	activity2	handover	occurrences	time
1	R2	R2	InformUser	ArchiveRepair	0.256951	268	2066
2	R2	R3	Register	AnalyzeDefect	0.259827	271	450
3	R2	R4	Register	AnalyzeDefect	0.249281	260	396
4	R2	R1	InformUser	ArchiveRepair	0.262704	274	1825
5	R3	R2	AnalyzeDefect	InformUser	0.255034	266	1950
6	R3	R2	TestRepair	ArchiveRepair	0.147651	154	2222
7	R3	R3	Repair(Simple)	TestRepair	0.225718	165	1821
8	R3	R3	Repair(Complex)	TestRepair	0.0593343	41	1998
9	R3	R3	TestRepair	RestartRepair	0.337731	128	1759
10	R3	R3	RestartRepair	Repair(Simple)	0.317857	89	1761
11	R3	R3	AnalyzeDefect	Repair(Complex)	0.128378	76	1068
12	R3	R3	AnalyzeDefect	Repair(Simple)	0.16408	74	1624
13	R3	R3	RestartRepair	Repair(Complex)	0.151515	15	1822
14	R3	R4	AnalyzeDefect	Repair(Complex)	0.155405	92	1480
15	R3	R4	TestRepair	RestartRepair	0.366755	139	1857
16	R3	R4	Repair(Simple)	TestRepair	0.168263	123	1873
17	R3	R4	AnalyzeDefect	Repair(Simple)	0.179601	81	1920

Fig. 5.3 Matrice del sociogramma

L'attributo "id" è l'identificativo delle riga della matrice, mentre "activity1" e "res1" indicano rispettivamente nome dell'attività di origine e la risorsa che l'ha eseguita, così come "activity2" e "res2" indicano lo stesso per l'attività di destinazione.

Ogni riga della matrice rappresenta un'etichetta del lato del sociogramma che va dal nodo origine "res1" al nodo destinazione "res2"; i nodi del sociogramma sono le risorse che hanno eseguito almeno un'attività nell'event log. Il campo "handover" contiene il valore dell'handover of work calcolato come spiegato nel capitolo 3. "Occurrences" è il numero di volte in cui "activity1" è stata eseguita prima di "activity2". "Time" indica il tempo

medio (in secondi) che passa dall'inizio dell'attività 1 all'inizio dell'attività 2. Per quandto riguarda invece il tempo medio di esecuzione di una specifica attività è stato calcolato come la media dei tempi dell'attività su tutte le tracce. Le attività e i loro tempi medi di esecuzione sono stati memorizzati nella tabella "activity", riportata in figura 5.4.

id	label	avgTime
1	Register	413
2	AnalyzeDefect	1673
3	Repair(Complex)	2076
4	TestRepair	1872
5	InformUser	2016
6	ArchiveRepair	1000
7	Repair(Simple)	1806
8	RestartRepair	1769

Fig. 5.4 Tempi medi di esecuzione delle attività, in secondi

Per come sono stati definiti e calcolati i tempi di esecuzione, non si è potuto stabilire quello dell'ultima attività, cioè Archive Repair, dato che tutte le tracce terminano al 100% con quest'attività (Fig. 4.2). Inoltre, per come è fatto l'event log del caso di studio, non si sarebbe potuto calcolare in nessun'altra maniera. Per questo si è deciso di assumere che il tempo medio dell'ultima attività è conosciuto a priori e viene dato in input all'applicazione. Sta di fatto che per ovviare a questo problema, in modelli di processo reali si potrebbe modificare la metrica per il calcolo dei tempi di esecuzione, ad es. calcolando la differenza tra il Complete Time e lo Start Time. In questo caso sarebbe stato inutile poiché, come già detto, il tempo di inizio e di fine per ogni evento erano identici. Tutto ciò comunque dipende dal contesto di applicazione e dalla struttura dell'event log.

5.2.2. Sostituzione della risorsa

La seconda parte dell'applicazione si occupa della sostituzione della risorsa mancante. A priori deve essere definita la risorsa (in un file di configurazione che sarà approfondito in 5.3) e deve essere stato estratto il sociogramma, memorizzato sul database. A questo punto l'applicazione ricava il valori di affinità tra la risorsa mancante e tutte le risorse candidate.

Ciò è stato fatto seguendo la metrica definita in 3.3.5. Di seguito la tabella "affinity" creata sul db.

id	resource	candidate	activityFactor	collaborationFactor	experience	speed	total
1	R2	R3	0	1	0	0	0.454545
2	R2	R4	0	1	0	0	0.454545
3	R2	R5	0	1	0	0	0.454545
4	R2	R1	1	1	0	0.333333	0.924242
5	R2	R6	0	1	0	0	0.454545

Fig. 5.5 Tabella "affinity" con i valori di affinità tra le risorse

Nella figura 5.5 si può vedere la tabella "affinity" per la risorsa R2. Il nome della risorsa mancante e di quella candidata si trovano rispettivamente nelle colonne "resource" e "candidate". Le altre colonne ad eccezione di "total" contengono i valori parziali nel calcolo della metrica, mentre "total" appunto è il valore finale dell'affinità, compreso tra 0 e 1 (1 è il massimo, 0 il minimo).

A questo punto nell'applicazione, è stato definito il modello matematico attraverso l'uso di CPLEX [12], un programma per la risoluzione di modelli matematici che fornisce una libreria Java.

Successivamente l'applicazione chiede in input la lista delle attività della risorsa mancante che si intende sostituire, che va inserita manualmente. Si può inserire più volte una stessa attività in quanto nel periodo di tempo considerato potrebbe essere stata programmata più volte, ma non è necessario inserire tutte le attività che è in grado di svolgere la risorsa.

In seguito viene creata una matrice dei costi c_i^j, calcolati secondo la definizione data il 3.3.3 e memorizzati nel db nella tabella "cost" mostrata nella figura sottostante.

id ▲ 1	resource	candidate	activity	collaboration	experience	speed	total
1	R2	R3	Register	0	0	0	1
2	R2	R3	InformUser	0	0	0	1
3	R2	R3	ArchiveRepair	0	0	0	1
4	R2	R4	Register	0	0	0	1
5	R2	R4	InformUser	0	0	0	1
6	R2	R4	ArchiveRepair	0	0	0	1
7	R2	R5	Register	0	0	0	1
8	R2	R5	InformUser	0	0	0	1
9	R2	R5	ArchiveRepair	0	0	0	1
10	R2	R1	Register	1	0.964218	1	0.0119272
11	R2	R1	InformUser	1	0.924354	0.931992	0.0478845
12	R2	R1	ArchiveRepair	0	1	0	0
13	R2	R6	Register	0	0	0	1
14	R2	R6	InformUser	0	0	0	1
15	R2	R6	ArchiveRepair	0	0	0	1

Fig. 5.6 Tabella "cost"

La figura 5.6 riporta i fattori di costo per la risorsa mancante "R2". Anche in questo caso sono stati comunque salvati i valori parziale del calcolo ed in "total" si ha il costo effettivo, con valori compresi tra 0 e 1 (rispettivamente minimo e massimo).

Per quanto riguarda i fattori di costo l_i (3.3.3) della risorsa r_i sono facilmente ottenibili come $l_i = 1 - beta \cdot (maxWorkload_i - currWorkload_i)$ dalla tabella "resource" in figura 5.7. Il coefficiente $beta$ serve a dare più o meno rilevanza alla disponibilità rispetto alla similarità e viene preso in input da un file di configurazione.

id	name	maxWorkload	currWorkload
1	R2	8	4
2	R3	8	2
3	R4	8	3
4	R5	8	0
5	R1	8	0
6	R6	8	0

Fig. 5.7 Tabella "resource"

Nella figura 5.7, il maxWorkload e il currWorkload che contengono rispettivamente il massimo carico di lavoro che la risorsa nella colonna "name" può eseguire ed il carico di lavoro corrente della risorsa. Il carico di lavoro per ogni risorsa nel caso di studio è stato preso come un valore casuale, ma a partire da uno scheduling $S(r)$ per le risorse (sequenza di attività da compiere in un periodo di tempo) può essere facilmente calcolato come $\sum_{i \in S(r)} avgTime(i)$ dove l'avgTime di ogni attività è contenuto nella tabella activity (fig. 5.4). Va specificato che in questo caso di studio il carico di lavoro è stato definito in ore, mentre i tempi di esecuzione in secondi, quindi viene effettuata la conversione in una misura comune (secondi). Una volta ottenuti tutti i coefficienti del modello matematico, viene avviato il risolutore CPLEX. I risultati vengono scritti in output in un file "replace.csv" creato dall'applicazione.

5.2 Configurazione

L’applicazione prevede una fase di configurazione molto semplice che consiste nello scrivere/modificare un file “config.txt” .

In questo file vanno definite le variabil per la connessione al database MySQL, che sono:

- username: l’username usato per la connessione al db;
- password: la password correlata all’username;
- dbName: nome del database;
- dbUrl: Url per la connessione al db, genericamente del tipo protocol//[hosts][/database][?properties];
- dbCreato: variabile boolean che indica se il db deve essere creato o meno (0 se non esiste, 1 altrimenti).

Oltre a questo vanno configurati i percorsi per i file di input e di output:

- graphsFile: nome del file e percorso in cui scrivere l'output di B.I.G.;
- eventLog: nome del file e percorso dell'event log;
- model: nome del file e percorso della rete di Petri;
- conformance: percorso della sotto directory di “folder” che deve chamarsi "Conformance", contenente i risultati del conformance checking (da definire anche se non viene effettuato);
- graphsFolder: percorso della sotto directory di”folder” che deve chamarsi "graphsDot", in cui vengono serializzati gli instance graphs;
- folder: percorso della directory.

Infine vanno definiti:

- lastActivity ed avgTime: nome e tempo medio di esecuzione dell'ultima attività;
- resource:il nome della risorsa da sostituire;
- w0, w1, w2, w3: pesi per il calcolo dell'affinità e della similarità;

  Nota: w0 è il peso che viene associato al parametro activity, che da una stima sulle attività in comune tra la risorsa mancante e quella candidata. Per un buon calcolo dell'affinità questo peso deve essere di almeno un ordine di grandezza maggiore rispetto a w2 e w3 (pesi che valutano l'esperienza e la velocità di esecuzione): infatti quest'ultimi si calcolano a partire dalle attività in comune (non avrebbe senso calcolare l'affinità su attività diverse). Ponendo tutti allo stesso livello si avrebbe che ad es. una risorsa che svolge una sola attività in comune con la risorsa mancante, ma che per quell'attività ha più esperienza ed è più veloce, risulterebbe migliore di una risorsa che ha tutte le attività in comune con la risorsa mancante ma per solo la metà di queste è più veloce e con più esperienza. Anche w1 che è peso relativo alle risorse comuni con cui collaborano, si consiglia di porlo allo stesso ordine di w0, perché è un parametro a cui si vuole dare più rilevanza per tenere in maggior considerazione l'aspetto sociale dell'azienda.
- beta: coefficiente dei costi di disponibilità.

6. Risultati

6.1 Caso di studio

Eseguendo l'applicazione dando in input i file contenenti l'event log ed il modello del processo del caso di studio (4), con la configurazione mostrata nella figura 6.1, i risultati ottenuti sono stati molto soddisfacenti.

```
#Percorsi per file di input e output (con doppio backslash)

#nome del file e percorso in cui scrivere l'output di BIG
graphsFile= C:\\Users\\ciott\\Desktop\\Tesi\\RepairF\\prova_graphs.g

#nome del file e percorso dell'event log
eventLog=C:\\Users\\ciott\\Desktop\\Tesi\\RepairF\\repair.xes

#nome del file e percorso della rete di Petri
model=C:\\Users\\ciott\\Desktop\\Tesi\\RepairF\\repair.pnml

#percorso della sotto directory di FolderName che deve chamarsi "Conformance"
conformance=C:\\Users\\ciott\\Desktop\\Tesi\\RepairF\\Conformance

#percorso della sotto directory di FolderName che deve chamarsi "graphsDot"
graphsFolder=C:\\Users\\ciott\\Desktop\\Tesi\\RepairF\\graphsDot\\

#percorso della directory
folder=C:\\Users\\ciott\\Desktop\\Tesi\\RepairF\\

#ultima attività e relativo tempo medio stimato a priori
lastActivity = ArchiveRepair
avgTime = 1000

#il nome della risorsa da sostituire
resource = R3

#pesi per il calcolo dell'affinità e della similarità
w0 = 10
w1 = 10
w2 = 1
w3 = 1

#coefficiente dei costi di disponibilità
beta = 1
```

Fig. 6.1 File di configurazione per l'esecuzione sul caso di studio.

Com'è possibile notare dalla figura 6.1, sono stati definiti i file di input, cioè "repair.xes" e "repair.pnlm", la risorsa da sostituire, l'ultima attività e tutti i pesi per la risoluzione del problema.

Il tempo di esecuzione della prima parte di dell'applicazione, ovvero quella che estrae il sociogramma, è mediamente di 300 secondi. Nella figura 6.2 viene riportato un esempio con il tempo di un esecuzione.

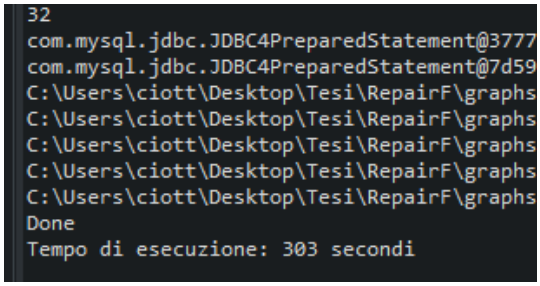


Fig 6.2 Tempo di esecuzione della prima fase.

Il sociogramma è stato creato in questa fase, e viene mostrato a titolo di esempio nella figura 6.3 oppurtunamente filtrato per rendere possibile la visualizzazione (sono stati resi visibili solo gli archi con un handover of work maggiore del 30%).

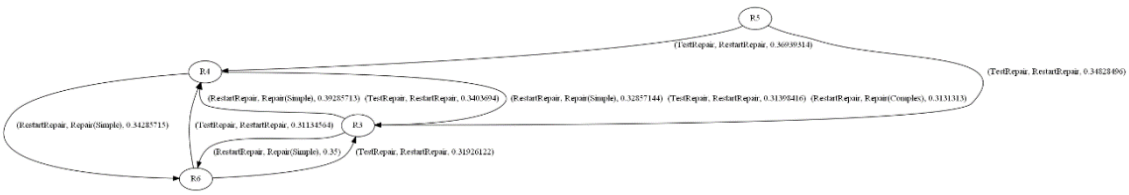


Fig. 6.3 Sociogramma filtrato con handover of work maggiore del 30%.

Per la seconda parte dell'applicazione, come si evince dalla figura 6.1, la risorsa scelta da sostituire è "R3". Di seguito viene mostrata la scelta della lista di attività da rimpiazzare.

```
Attività che svolge R3:
AnalyzeDefect
Repair(Complex)
TestRepair
Repair(Simple)
RestartRepair
Inserisci le attività per cui R3 deve essere sostituito ('ok' per terminare):
AnalyzeDefect
TestRepair
RestartRepair
RestartRepair
ok
```

Fig. 6.4 Inserimento della lista delle attività.

Si vede dalla figura 6.4 che una stessa attività può essere inserita più volte (es. Restart Repair). Per queste attività e questa risorsa l'applicazione ha restituito i seguenti risultati:

```
Found incumbent of value 4.000000 after 0.02 sec. (0.00 ticks)
Tried aggregator 2 times.
MIP Presolve eliminated 5 rows and 16 columns.
MIP Presolve modified 1 coefficients.
Aggregator did 4 substitutions.
All rows and columns eliminated.
Presolve time = 0.01 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.03 sec. (0.02 ticks)
Parallel b&c, 4 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.03 sec. (0.02 ticks)
Solution status = Optimal
Solution value= 0.005307789891958237
Done
Tempo di esecuzione: 21 secondi
```

Fig. 6.5 Risoluzione del modello matematico e valori della soluzione.

Nella figura 6.5 è possibile vedere come il risolutore trova la soluzione ottima in 0.03 secondi ed il tempo di esecuzione è di 21 secondi, ma dipende dal tempo d'inserimento delle attività quindi si può considerare come immediato.

Risorsa da sostituire: R3					
Risorsa Candidata	ActivityFactor	CollaborationFactor	Experience	Speed	Affinità
R4	1	1	1	0.4	97%
R6	0.6	1	1	1	81%
R5	0.6	1	1	0.666667	80%
R2	0	0.666667	0	0	30%
R1	0	0.666667	0	0	30%
Candidato	AnalyzeDefect	TestRepair	RestartRepair	RestartRepair	
R2	0	0	0	0	
R4	1	1	1	1	
R5	0	0	0	0	
R1	0	0	0	0	
R6	0	0	0	0	

Fig. 6.6 Soluzione del problema.

Nella figura 6.6 è mostrata la soluzione del problema. Si evince che la risorsa migliore per sostituire R3 è R4 con il 97% di affinità, ciò è confermato dal fatto che la stessa R4 viene scelta per sostituire tutte le attività di R3. I risultati sono conformi a quanto stabilito a priori (4.3). R3 fa parte del team operativo composto anche da R4, R5, R6, con cui ha più affinità rispetto ad R2 ed R1 (che fanno parte dell'altro team). Inoltre ad R3 ed R4 era stato assegnato un ruolo più importante rispetto agli altri membri del team in quanto solo loro potevano svolgere le attività di Analyze Defect e Restart Repair. Questi risultati sono stati ottenuti coi i seguenti carichi di lavoro (figura 6.7):

name	maxWorkload	currWorkload
R2	8	6
R3	8	2
R4	8	3
R5	8	6
R1	8	7
R6	8	5

Fig. 6.7 Carichi di lavoro delle risorse.

Il workload massimo per ogni risorsa è pari ad 8 ore ed dalla figura 6.7 si nota come R4 abbia 5 ore disponibili. Modificando il suo carico di lavoro corrente ad 8 ore, quindi definendo R4 come una risorsa non disponibile, ci si aspetta che siano scelte altre risorse per rimpiazzare R3. Infatti riavviando l'applicazione per le stesse attività, questo viene verificato e nella figura 6.8 è possibile vedere che vengono scelti R5 ed R6.

Candidato	AnalyzeDefect	TestRepair	RestartRepair	RestartRepair
R2	0	0	0	0
R4	0	0	0	0
R5	1	0	0	1
R1	0	0	0	0
R6	0	1	1	0

Fig. 6.8 Soluzione per R4 non disponibile.

Questo però fa aumentare notevolmente il valore della funzione obiettivo che passa da 0.0053 (figura 6.5) ad un valore pari a 3 (figura 6.9).

```
Root node processing (before b&c):
  Real time             =    0.03 sec. (0.02 ticks)
Parallel b&c, 4 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.03 sec. (0.02 ticks)
Solution status = Optimal
Solution value= 3.0
```

Fig. 6.9 Valore della soluzione con R4 non disponibile.

Vengono riportate nelle figure seguenti le risoluzioni del modello per le altre risorse su tutte le attività che sono in grado di svolgere (considerate una sola volta).

Risorsa da sostituire: R1					
Candidato	ActivityFactor	CollaborationFactor	Experience	Speed	Totale
R2	1	1	0.666667	0.333333	95%
R3	0	1	0	0	45%
R4	0	1	0	0	45%
R5	0	1	0	0	45%
R6	0	1	0	0	45%
Candidato	ArchiveRepair	Register	InformUser		
R2	1	1	1		
R3	0	0	0		
R4	0	0	0		
R5	0	0	0		
R6	0	0	0		

Fig. 6.10 Soluzione del modello per R1

Risorsa da sostituire: R2					
Candidato	ActivityFactor	CollaborationFactor	Experience	Speed	Totale
R1	1	1	0	0.333333	92%
R3	0	1	0	0	45%
R4	0	1	0	0	45%
R5	0	1	0	0	45%
R6	0	1	0	0	45%
Candidato	Register	InformUser	ArchiveRepair		
R3	0	0	0		
R4	0	0	0		
R5	0	0	0		
R1	1	1	1		
R6	0	0	0		

Fig. 6.11 Soluzione del modello per R2

Risorsa da sostituire: R3					
Candidato	ActivityFactor	CollaborationFactor	Experience	Speed	Totale
R4	1	1	1	0.4	92%
R6	0.6	1	1	1	81%
R5	0.6	1	1	0.666667	80%
R2	0	0.666667	0	0	30%
R1	0	0.666667	0	0	30%
Candidato	AnalyzeDefect	Repair(Complex)	TestRepair	Repair(Simple)	RestartRepair
R2	0	0	0	0	0
R4	1	0	0	0	1
R5	0	1	1	0	0
R1	0	0	0	0	0
R6	0	0	0	1	0

Fig. 6.12 Soluzione del modello per R3

Risorsa da sostituire: R4					
Candidato	ActivityFactor	CollaborationFactor	Experience	Speed	Totale
R3	1	1	0.2	0.6	94%
R5	0.6	1	0.333333	1	78%
R6	0.6	1	0	1	77%
R2	0	0.666667	0	0	30%
R1	0	0.666667	0	0	30%
Candidato	Repair(Complex)	AnalyzeDefect	Repair(Simple)	RestartRepair	TestRepair
R2	0	0	0	0	0
R3	0	1	0	1	0
R5	1	0	1	0	0
R1	0	0	0	0	0
R6	0	0	0	0	1

Fig. 6.13 Soluzione del modello per R4

Risorsa da sostituire: R5					
Candidato	ActivityFactor	CollaborationFactor	Experience	Speed	Totale
R4	1	1	1	0.666667	98%
R6	1	1	0.333333	0.666667	95%
R3	1	1	0	0.333333	92%
R2	0	0.666667	0	0	30%
R1	0	0.666667	0	0	30%
Candidato	TestRepair	Repair(Simple)	Repair(Complex)		
R2	0	0	0		
R3	0	0	0		
R4	0	1	1		
R1	0	0	0		
R6	1	0	0		

Fig. 6.14 Soluzione del modello per R5

Risorsa da sostituire: R6					
Candidato	ActivityFactor	CollaborationFactor	Experience	Speed	Totale
R3	1	1	0.333333	0.666667	95%
R4	1	1	1	0	95%
R5	1	1	0.666667	0.333333	95%
R2	0	0.666667	0	0	30%
R1	0	0.666667	0	0	30%
Candidato	TestRepair	Repair(Simple)	Repair(Complex)		
R2	0	0	0		
R3	0	0	1		
R4	0	0	0		
R5	1	1	0		
R1	0	0	0		

Fig. 6.15 Soluzione del modello per R6

6.2 Processo reale

Dato che erano stati ottenuti i risultati attesi per il caso di studio, per verificare l'effettiva validità del lavoro si è pensato di testare l'applicazione su un event log e un modello di processo reale, in modo da valutare la scalabilità per delle mole di dati più rilevanti. In particolare è stato usato un event log preso da una BPIC (Business Process Intelligence Challenge) [13]. Questo log è stato ottenuto da un istituto finanziario olandese e contiene 63.023 eventi in 7974 case. A parte qualche dato anonimizzato, il log contiene tutti i dati così come provengono dall'istituto finanziario. Il processo rappresentato nell'event log riguarda la domanda fatta per un prestito personale o per uno scoperto all'interno dell'organizzazione finanziaria globale. Il log è una fusione di tre sottoprocessi intrecciati. La prima lettera di ciascun nome di attività identifica da quale processo secondario ha avuto origine. Di seguito le informazioni sul log del processo.

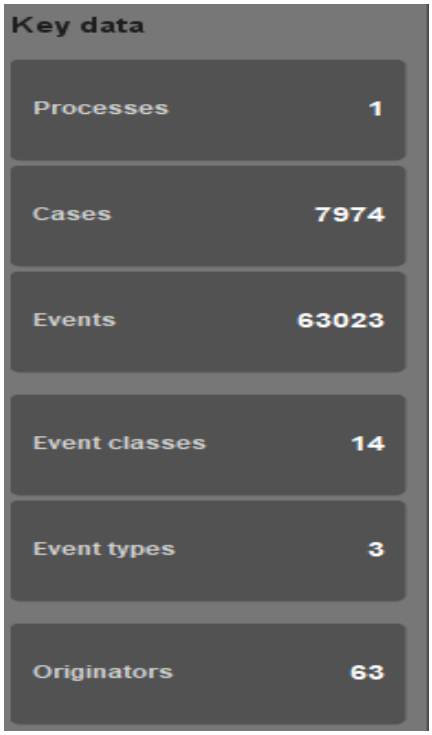


Fig. 6.16 Informazioni sull'event log del processo

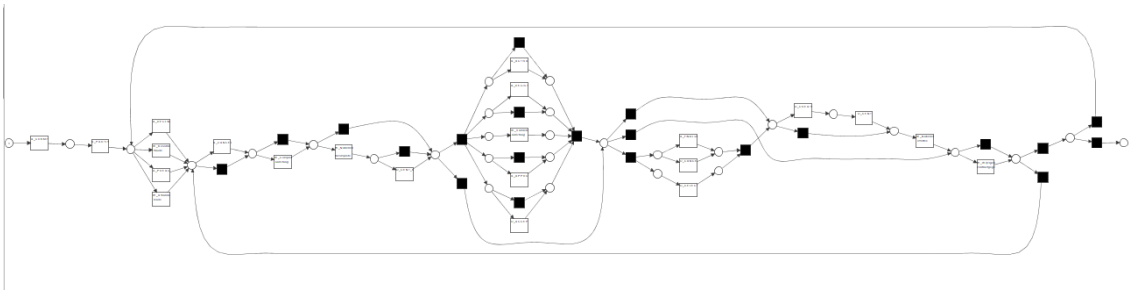


Fig. 6.17 Rete di Petri del processo

Dalla figura 6.16 si nota come gli originators cioè le risorse che eseguono le attività durante l'esecuzione del processo sono 63 e i tipi di eventi sono 14. Anche dal modello del processo (figura 6.17) risulta chiaro di come la complessità di questo processo sia maggiore rispetto al caso di studio analizzato in precedenza. In questa situazione, avere

uno strumento di supporto per il replacement planning risulta essere un importante aiuto. Di seguito riportiamo un'esempio dell'esecuzione dell'applicazione sul processo reale, sebbene siano stati fatti numerosi test.

L'applicazione per l'esempio è stata configurata nel modo illustrato dalla figura 6.18, scegliendo di sostituire la risorsa "112".

```
#Percorsi per file di input e output (con doppio backslash)

#nome del file e percorso in cui scrivere l'output di BIG
graphsFile= C:\\Users\\ciott\\Desktop\\Tesi\\BPI\\prova_graphs.g

#nome del file e percorso dove Ã¨ l'event log
eventLog=C:\\Users\\ciott\\Desktop\\Tesi\\BPI\\bpi.xes

#nome del file e percorso dove Ã¨ la rete di Petri
model=C:\\Users\\ciott\\Desktop\\Tesi\\BPI\\bpi.pnml

#percorso della sotto directory di FolderName che deve chamarsi "Conformance"
conformance=C:\\Users\\ciott\\Desktop\\Tesi\\BPI\\Conformance

#percorso della sotto directory di FolderName che deve chamarsi "graphsDot"
graphsFolder=C:\\Users\\ciott\\Desktop\\Tesi\\BPI\\graphsDot\\

#percorso della directory
folder=C:\\Users\\ciott\\Desktop\\Tesi\\BPI\\

#ultima attività e relativo tempo medio stimato a priori
lastActivity = W_Afhandelen leads
avgTime = 1000

#il nome della risorsa da sostituire
resource = 112

#pesi per il calcolo dell'affinità e della similarità;
w0 = 10
w1 = 10
w2 = 1
w3 = 1
#coefficiente dei costi di disponibilità
beta = 1
```

Fig. 6.18 File di configurazione per il processo reale

La fase di estrazione del sociogramma per questo event log e questo modello ha necessitato di molto più tempo rispetto ai 5 minuti del caso di studio. Nello specifico ha impiegato 41766 secondi cioè circa 11 ore e 36 minuti (figura 6.19).

```
C:\Users\ciott\Desktop\Tesi\BPI\graphsDot\matrix.dot
C:\Users\ciott\Desktop\Tesi\BPI\graphsDot\matrix_10perc.dot
C:\Users\ciott\Desktop\Tesi\BPI\graphsDot\matrix_15perc.dot
C:\Users\ciott\Desktop\Tesi\BPI\graphsDot\matrix_20perc.dot
C:\Users\ciott\Desktop\Tesi\BPI\graphsDot\matrix_25perc.dot
Done
Tempo di esecuzione: 41766 secondi
```

Fig. 6.19 Tempo di esecuzione per l'estrazione del sociogramma.

La fase di replacement invece è rapida anche per grandi quantità di dati.

```
Attività che svolge 112:
A_SUBMITTED
A_PARTLYSUBMITTED
A_DECLINED
A_PREACCEPTED
W_Completerenaanvraag
W_Afhandelenleads
A_CANCELLED
W_Beoordelenfraude
Inserisci le attività per cui 112 deve essere sostituito ('ok' per terminare):
A_SUBMITTED
A_PREACCEPTED
W_Completerenaanvraag
W_Beoordelenfraude
ok
Tried aggregator 1 time.
MIP Presolve eliminated 65 rows and 244 columns.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.13 ticks)

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.15 ticks)
Parallel b&c, 4 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.15 ticks)
Solution status = Optimal
Solution value= 2.023965612053871
Done
Tempo di esecuzione: 90 secondi
```

Fig. 6.20 Esecuzione della fase di replacing.

Si può vedere nella figura 6.20 come il tempo di esecuzione sia di 90 secondi cioè poco più grande rispetto al caso di studio. In ogni caso il tempo di calcolo del risolutore è pressochè identico a prima (0.02 secondi).

Questa volta, nell'esecuzione esempio le attività scelte sono state A_SUBMITTED, A_PREACCEPTED, W_Completerenaanvraag e W_Beoordelenfraude. I risultati per queste attività e la risorsa 112 sono stati i seguenti.

Risorsa da sostituire: 112					
Risorsa Candidata	ActivityFactor	CollaborationFactor	Experience	Speed	Affinità
11169	0.75	0.8	0.166667	0.333333	72%
10909	0.75	0.78	0.166667	0.333333	71%
11179	0.75	0.72	0.166667	0.333333	69%
11003	0.75	0.7	0.166667	0.333333	68%
10861	0.625	0.8	0.2	0.2	66%
11180	0.625	0.76	0.2	0.2	64%
11203	0.625	0.76	0.2	0.2	64%
10932	0.75	0.62	0.166667	0.333333	64%
10982	0.625	0.74	0.2	0.2	63%
11122	0.625	0.74	0.2	0.2	63%
10913	0.625	0.74	0.2	0.2	63%
11119	0.625	0.72	0.2	0.2	62%
11181	0.625	0.7	0.2	0.4	62%
11201	0.625	0.7	0.2	0.2	62%
10929	0.75	0.56	0.166667	0.333333	61%
11189	0.625	0.66	0.2	0.4	61%
11121	0.625	0.66	0.2	0.2	60%
11302	0.75	0.46	0.166667	0.833333	59%
10863	0.625	0.64	0.2	0.2	59%

Fig. 6.21 Risorse con maggiore affinità

Candidato	A_SUBMITTED	A_PREACCEPTED	W_Completerenaanvraag	W_Beoordelenfraude
10912	1	0	0	0
11111	0	0	0	0
10982	0	0	0	0
11019	0	0	0	0
11180	0	0	0	0
10939	0	0	0	0
11000	0	0	0	0
11201	0	0	1	0
11169	0	0	0	0
11179	0	0	0	0

Fig.6.22 Soluzione del modello (parte 1)

10859	0	1	0	0
10228	0	0	0	0
10899	0	0	0	0

Fig. 6.23 Soluzione del modello (parte 2)

11309	0	0	0	0
11304	0	0	0	1

Fig. 6.24 Soluzione del modello (parte 3)

I risultati sono stati riportati in più figure (fig. 6.21, 6.22, 6.23 e 6.24) poiché era impossibile racchiuderli in un'unica immagine. In questo caso più complesso viene alla luce come non sempre le risorse più affini sono la soluzione ottimale del problema. Ciò è normale in quanto l'affinità è una misura generale tra le risorse mentre ogni attività va considerata nel caso specifico. Inoltre la disponibilità diventa un peso più rilevante nel calcolo del problema. Comunque i risultati ottenuti sono soddisfacenti poichè è stato controllato che le risorse scelte per le attività normalmente (sia nell'esempio riportato, sia negli altri test effettuati) sono quelle che hanno i costi relativi minori. Ad es. la risorsa "11201" scelta per l'attività di "W_Completerenaanvraag" ha il minor costo fra tutte le risorse. Come mostrato dalla figura 6.25.

```
SELECT * FROM `cost` WHERE activity = 'W_Completerenaanvraag' ORDER BY `cost`.`total` ASC
```

Profiling [Modifica inline] [Modifica] [Spiega SQL

1 > >> | Mostra tutti | Numero di righe: 25 Filtra righe: Cerca nella tabella Ordina per chiave: Nessuno

Opzioni

	id	resource	candidate	activity	collaboration	experience	speed	total ▲ 1
Modifica Copia Elimina	61	112	11201	W_Completerenaanvraag	0.772727	1	0.974821	0.0841506
Modifica Copia Elimina	93	112	11181	W_Completerenaanvraag	0.772727	1	0.927456	0.099939
Modifica Copia Elimina	141	112	11203	W_Completerenaanvraag	0.818182	1	0.863325	0.106164

Fig 6.25 Tabella dei costi ordinata dal minore al maggiore (per l'attività W_Completerenaanvraag)

6.3 Analisi dei risultati

L'ottimalità della soluzione è confermata sia dal risolutore del modello sia dalle verifiche fatte. Per quanto riguarda i tempi di esecuzione, la fase di replacement ha tempi brevissimi, per questo si potrebbe considerare anche un'applicazione run time. Per la fase di estrazione del grafo sociale invece si nota chiaramentte che i tempi scalano in maniera esponenziale. Non è stata fatta un'approfondita analisi di complessità ma ciò deriva sicuramente dal fatto che la creazione del grafo dipende dalla complessità del modello di processo, dal numero di risorse e dal numero di tracce. Per ovviare a questo problema si potrebbe pensare di parallelizzare la creazione della matrice suddividendo il log in n sottoinsiemi di tracce, per ciascun sottoinsieme estrarre la matrice di handover e infine sommare tutte le n matrici per ottenere il grafo completo. Si potrebbe anche considerara una finestra temporale di riferimento in maniera tale da mantenere la dimensione dell'event log constante, con sempre gli n sottoinsiemi di tracce più recenti. Al momento comunque il software risulta eseguibile nel contesto pensato per la sua applicazione considerando che la fase di estrazione del sociogramma è un'attività che va eseguita di rado. Una volta ottenuto il sociogramma, finchè non cambiano sostanzialmente i dati degli eventi, la fase di replacement può essere eseguita tranquillamente numerose volte.

7. Conclusione e sviluppi futuri

Questo studio di tesi mirava ad ottenere un strumento di replacement planning al fine di individuare correttamente la miglior soluzione per sostituire una risorsa mancante. L'obiettivo si può considerare raggiunto in quanto i risultati ottenuti mostrano chiaramente come l'applicazione è in grado di fornire una distribuzione corretta delle attività da sostituire e un'identificazione immediata della persona migliore per sopperire a quella non disponibile. Utilizzando infatti un caso di studio con dati artificiali, inseriti appositamente con l'aspettativa di ottenere certi risultati, è stato possibile constatare che l'applicazione ha confermato le attese. Ciò indica che la metodologia utilizzata per questo lavoro è valida, certamente perfezionabile, ma che sicuramente non conduce a risultati errati. L'utilizzo delle tecniche di organizational mining ci garantisce una maggior affidabilità perché la soluzione viene costruita a partire dai dati storici. La costruzione del grafo a partire dagli instance graph è inoltre un elemento di innovazione. Nello stato dell'arte infatti, la matrice di handover viene costruita a partire dalla tracce dell'event log. Questa modalità di costruzione però presenta delle incongruenze sulle relazioni di precedenza tra le attività del processo. Quindi la metodologia qui proposta per l'estrazione del sociogramma viene considerata come un'alternativa più valida. Comunque è importante sottolineare che l'applicazione implementata va adattata al contesto applicativo e non è un modello generale valido per tutte le realtà. La contestualizzazione è necessaria per l'effettivo funzionamento all'interno delle aziende. Gli sviluppi futuri che verranno proposti in questa sezione riguardono sia possibili migliorie dell'applicazione software, sia eventuali sviluppi della metodologia per arrivare a soluzioni innovative. Innanzitutto, per la parte software si propone di realizzare un

interfaccia utente che sia facile accesso per l'inserimento dei parametri di input. Infatti sarebbe utile implementare una soluzione grafica che permetta l'inserimento della risorsa da sostituire, la lista delle attività ed i pesi per le metriche. In questo modo si migliora notevolmente l'usabilità dell'applicazione, rendendola accessibile anche agli utenti meno esperti. Si potrebbe anche pensare di prendere un ulteriore file in input, contenente lo scheduling di ogni risorsa del processo. In questo modo non ci sarebbe bisogno di interagire con l'utente per reperire la lista delle attività. Inoltre si avrebbe una misura corretta del carico di lavoro corrente per ogni risorsa, rendendo più efficace l'applicazione. Un'ultima proposta per il software è quella di ottimizzare la fase di estrazione del sociogramma in maniera tale da renderla più veloce anche per grandi quantità di dati, come discusso in 6.3. Al di là del lato software, si potrebbe sviluppare ulteriormente il lavoro migliorandone le metriche oppure addirittura proponendo un modello matematico più efficiente. Il tutto andrebbe testato su processi di cui si conosce già la struttura per evidenziare l'andamento di un modello/metrica rispetto ad un altro/a. Oltre all'aiuto nel replacement planning, l'estrazione del sociogramma pone la base per lo sviluppo di nuove attività di organizational mining: si potrebbero mettere in evidenza altre informazioni utili per l'improvement del processo. Ad esempio rilevare quelle risorse sono sovraccariche o scoprire eventuali colli di bottiglia del sistema. È possibile anche sviluppare un metodo per fornire delle metriche di valutazione del lavoro svolto dalle risorse.

In conclusione, si auspica che la presente tesi sia un punto di partenza per la ricerca e lo sviluppo nell'ambito dell'organizational mining e dell'analisi delle reti sociali aziendali, vista l'utilità, i possibili campi di applicazione ed i miglioramenti che si riescono ad apportare ai processi.

Riferimenti

Sitografia

[1] Grandi aziende, www.staffroster.com/grandi-aziende/,

Pagina acceduta il 25 giugno 2019

[2] Process mining,

https://web.archive.org/web/20070504091600/http:/is.tm.tue.nl/staff/wvdaalst/BPMcenter/process%20mining.htmil,

Pagina acceduta il 27 giugno 2019.

[4] Rete di Petri, https://it.wikipedia.org/wiki/Rete_di_Petri,

Pagina acceduta il 3 luglio 2019.

[6] ProM, http://www.promtools.org/doku.php.

[7] Alpha Algorithm, https://en.wikipedia.org/wiki/Alpha_algorithm,

Pagina acceduta il 3 luglio 2019.

[9] Java, https://it.wikipedia.org/wiki/Java_(linguaggio_di_programmazione).

[10] MySQL, https://it.wikipedia.org/wiki/MySQL.

[12] Cplex, https://www.ibm.com/it-it/products/ilog-cplex-optimization-studio

[13] BPIC, https://www.win.tue.nl/bpi/doku.php?id=2012:challenge

Bibliografia

[3] Wil M.P. van der Aalst*, Process Mining Discovery, Conformance and Enhancement of Business Processes,* Springer, 2011

[5] Jie Tao, Amit V. Deokar, *An Organizational Mining Approach Based on Behavioral Process Patterns,* Twentieth Americas Conference on Information Systems, Savannah, 2014

[8] Wil MP Van Der Aalst, Hajo A Reijers, and Minseok Song*, Discovering social networks from event logs. Computer Supported Cooperative Work (CSCW)*, 14(6):549–593, 2005.

[11] Claudia Diamantini, Laura Genga, Domenico Potena, Wil van der Aalst, *Building instance graphs for highly variable processes,* Elsevier Ltd., Aprile 2016

Ringraziamenti

A conclusione di questo lavoro, mi sento in dovere di spendere qualche parola per ringraziare pubblicamente le persone che mi hanno dato il loro aiuto in questi mesi: se nella tesi c'è qualcosa di buono lo si deve principalmente a loro (mentre le mancanze sono ovviamente tutte mie).

Un sincero grazie al Prof. Domenico Potena ed al Dott. Emanuele Storti che mi hanno assistito e consigliato durante lo svolgimento di questa tesi e sono stati sempre presenti e disponibili nei miei confronti: i loro suggerimenti sono stati preziosissimi.

Ci tengo inoltre a ringraziare soprattutto mia madre, in particolare per gli sforzi fatti per contribuire ai miei studi. Spero di averla resa orgogliosa.

Un ringraziamento speciale va anche alla mia fidanzata Maria, a cui dedico questa tesi. È stata fondamentale per me: mi aiutato e sopportato, ha condiviso con me le ansie e mi ha sempre sostenuto e incoraggiato. Volevo ringraziare anche tutta la sua famiglia che mi ha accolto e che durante questi mesi di lavoro ed anni di studi è stata come una seconda casa.

Grazie anche a tutti i miei amici che sono stati sempre presenti e hanno reso speciale questo percorso. Se riuscirò a laurearmi, sarà in parte anche grazie a loro.