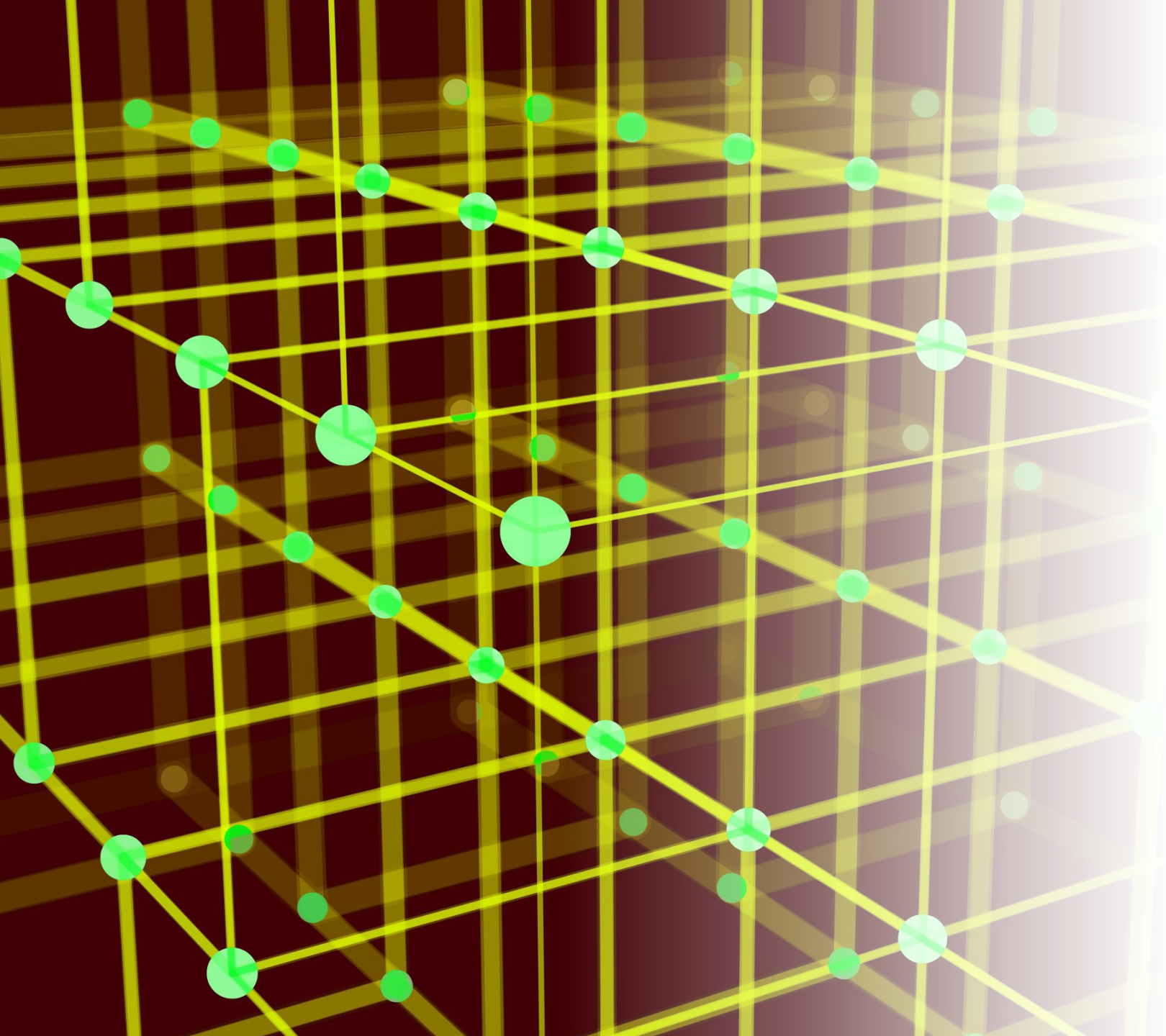


Subdue compression with LPM

Progetto di Big data analytics e Machine learning
UNIVPM A.A. 2021/2022

Obiettivo del progetto

L'obiettivo del seguente progetto è di realizzare un algoritmo di compressione che operi con la stessa logica di ***Subdue***, accettando in input *file di log* in formato *.xes* anziché grafi.

Il formato *.xes:*

- *È un* derivato del formato *.xml;*
- Contiene le tracce di un processo.

Introduzione

Subdue è un algoritmo di clustering gerarchico e concettuale, dove ogni cluster è descritto da un'etichetta e diviso in sotto-cluster.

Subdue comprime un grafo, sostituendo tutte le istanze di una sotto-struttura con un puntatore alla struttura stessa.

Introduzione

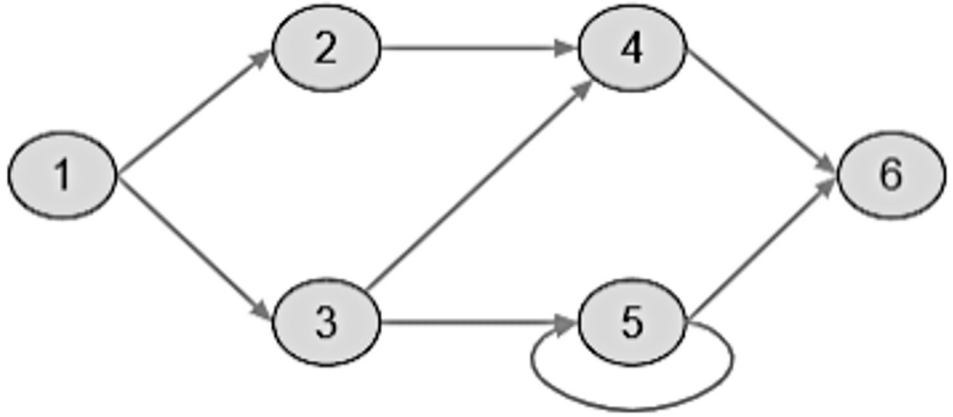
Per determinare il sotto-grafo candidato, il criterio di scelta di *Subdue* si basa sul principio della ***Minimum Description Length*** (***MDL***):

$$Compression = \frac{DL(S) + DL(G|S)}{DL(G)}$$

- $DL(G)$ è la *Description Length* del grafo di input;
- $DL(S)$ è la *Description Length* della sotto-struttura;
- $DL(G|S)$ è la *Description Length* del grafo di input dal quale sono state eliminate tutte le istanze della sotto-struttura.

Introduzione

La *Description Length* di un grafo si calcola come il numero di bit necessari per codificare la *matrice di adiacenza*, ovvero la struttura dati che implementa il grafo.
La struttura scelta è quella che massimizza l'inverso di *Compression*.



nodi di partenza	1	2	3	4	5	6
1	0	1	1	0	0	0
2	0	0	0	1	0	0
3	0	0	0	1	1	0
4	0	0	0	0	0	1
5	0	0	0	0	1	1
6	0	0	0	0	0	0

Considerazioni su Subdue

Come appena visto, ***Subdue*** accetta in input un grafo, perciò non si adatta al nostro obiettivo; quindi, è stato sviluppato un nuovo algoritmo ispirato a ***Subdue***, con le seguenti accortezze:

- I dati in ingresso devono essere un *file di log ed* ed i *LPM* precedentemente ottenuti;
- La metrica di compressione deve tener conto del concetto di *traccia*;
- Il processo di sostituzione deve eliminare dal *file di log* gli eventi della sotto-struttura di interesse, sostituendo il primo evento della traccia valida con un'etichetta.

Workflow

01

Generazione dei LPM

02

Etichettatura degli eventi

03

Scelta del LPM

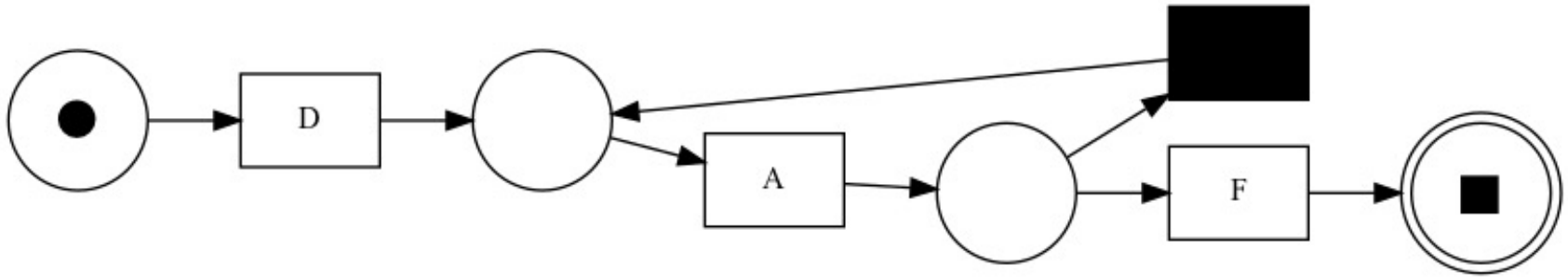
04

Eliminazione del LPM candidato

Generazione del LPM

Un ***Local Process Model*** è una *rete di Petri* che identifica e descrive un comportamento molto frequente all'interno *dell'event log.*

Tramite il framework ***ProM***[1] sono stati generati i *LPM* da fornire in input all'algoritmo di etichettatura.



Etichettatura degli eventi

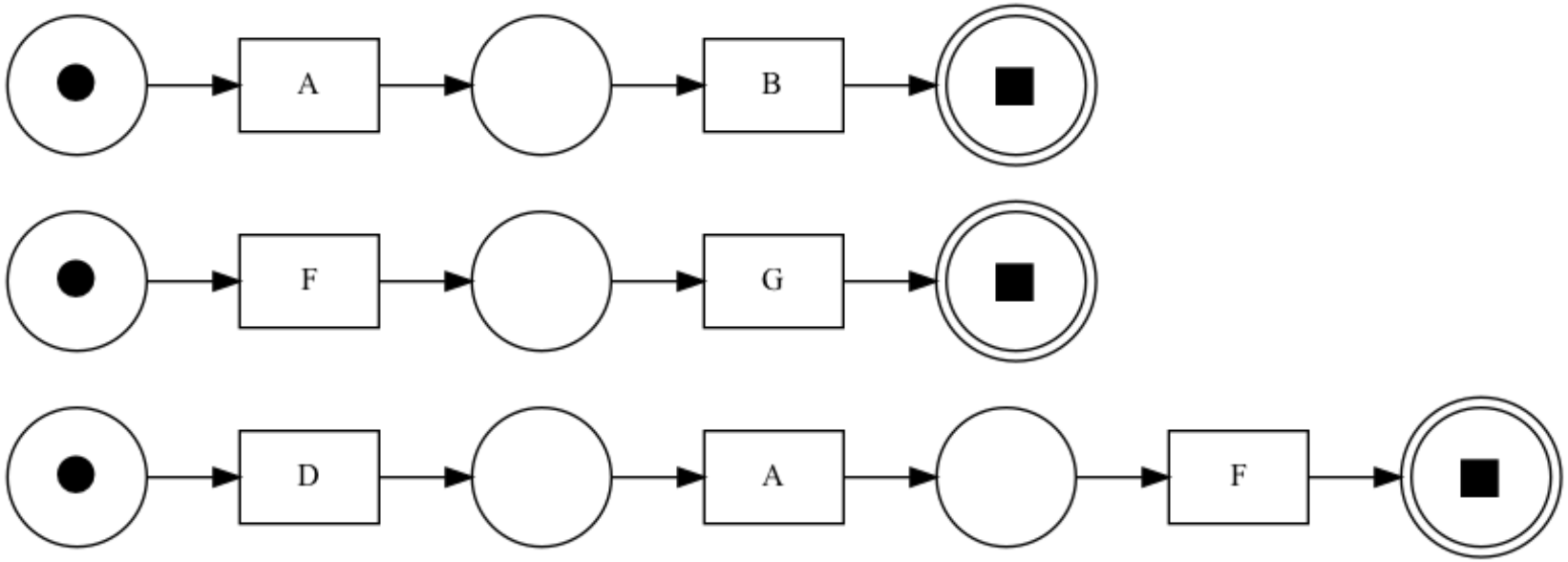
L'algoritmo di etichettatura è sintetizzabile nei seguenti passi:

- Riceve in input il *file di log* in formato *.xes* ed i *LPM* precedentemente ottenuti;
- Restituisce in output il file *.xes* originale, dove ad ogni evento viene assegnata un lista contente i *LPM* in cui l'evento compare;

L'evento, per poter essere etichettato, deve comparire in una traccia che percorre interamente il LPM dallo stato iniziale a quello finale (traccia valida), altrimenti viene ignorato.

Esempio

$$L = [< a,b,c,d,a,b >,< a,c,d,a >,< f,g,h,i >,< d,a,f,g >]$$



$$L = [< a(1),b(1),c,d,a(1),b(1) >,< a,c,d,a >,< f(2),g(2),h,i >,< d(3),a(3),f(2,3),g(2) >]$$

Scelta del LPM

Per poter sostituire gli eventi nel *file di log*, è necessario definire una metrica per la compressione, alcune prese in considerazione sono:

- Numero di eventi contrassegnati con il *LPM*: calcolata come il numero di eventi in cui occorre l'indice relativo al *LPM*;
- Numero di tracce che percorrono il *LPM* per il numero di transizioni dello stesso; è il metodo che maggiormente si avvicina alla metrica utilizzata da *Subdue e* si calcola come:

$$N° \; tracce \; * Dimensione \; LPM$$

Scelta del LPM

In output dall'algoritmo di etichettatura otteniamo gli eventi che sono contrassegnati con gli indici dei LPM in cui compaiono in una traccia valida, perciò la prima metrica è più facile da calcolare, poiché basta scorrere il *file di log* e selezionare il LPM il cui indice compare il maggior numero di volte.

Inoltre, questa metrica definisce esattamente il numero degli eventi che verranno sostituiti, per cui si adatta meglio all'obiettivo del progetto.

Eliminazione del LPM

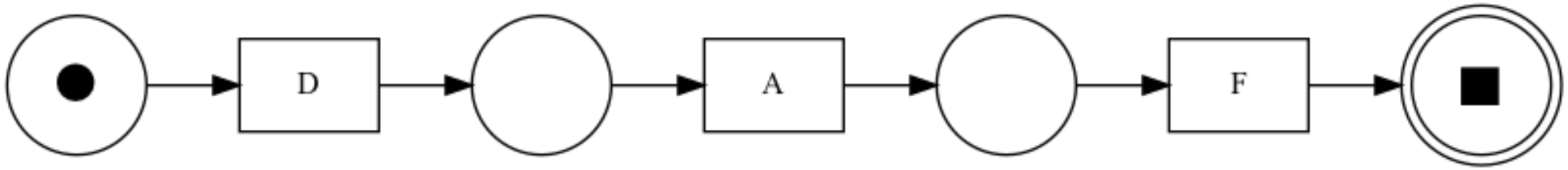
Dopo aver individuato il *LPM* candidato, si comprime il *file di log* scorrendolo e manipolando gli eventi:

- Se l'evento corrente è il primo evento di una traccia valida per il *LPM* candidato, allora viene sostituito da un evento con il nome: «*LPM:*numLPM*_*num*Iterazione*», altrimenti viene eliminato;
- Inoltre, è necessario eliminare gli attributi «*LPMs_List*», «*LPMs_Binary*», «*LPMs_Frequency*» dell'evento, perché verrà etichettato di nuovo nella successiva iterazione.

Eliminazione del LPM: 1

Esempio 1

$$\mathrm{L} = [< d(1), a(1), f(1,2), g(2), d(1), a(1), f(1,2) >]$$



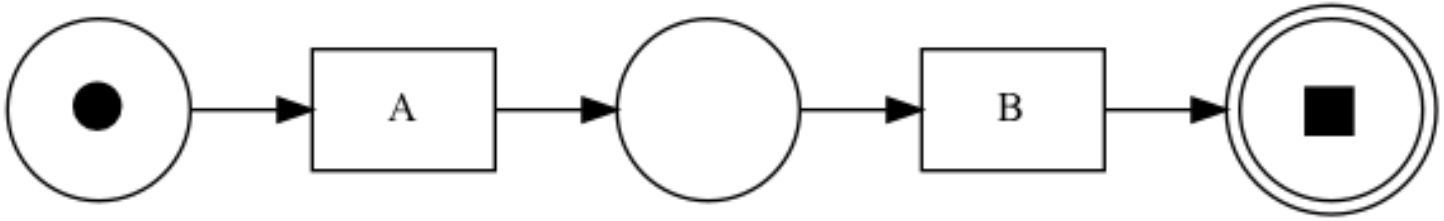
$$Initial = [\mathrm{D}],\ final = [\mathrm{F}]$$

$$\mathrm{L} = [< LPM{:}\,1\_0, g, LPM{:}\,1\_0 >]$$

Eliminazione del LPM: 1

Esempio 2

$$\mathrm{L} = [< a(1), b(1), a(1), b(1), c, a(1), b(1) >]$$



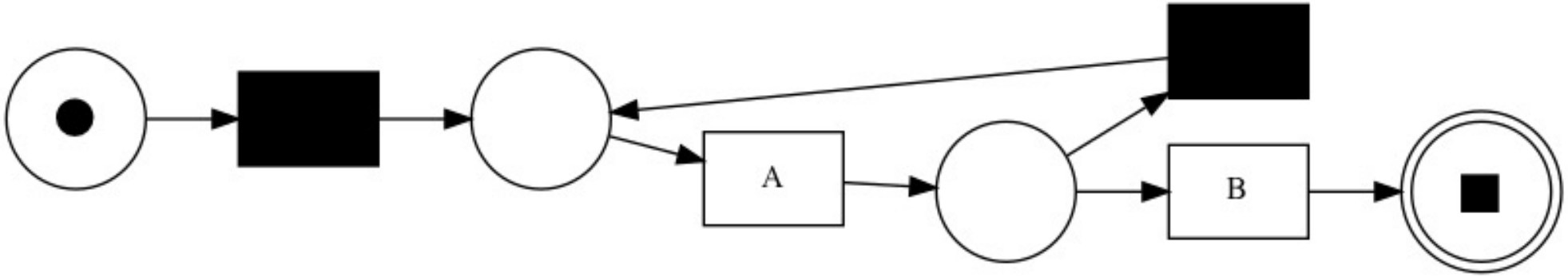
$$\mathit{Initial} = [\mathrm{A}], \mathit{final} = [\mathrm{B}]$$

$$\mathrm{L} = [< \mathit{LPM}\mathrm{:1\_0}, \mathit{LPM}\mathrm{:1\_0}, c, \mathit{LPM}\mathrm{:1\_0} >]$$

Eliminazione del LPM: 1

Esempio 3

$$\mathrm{L} = [< a(1), b(1), a(1), b(1), c, a(1), b(1) >]$$



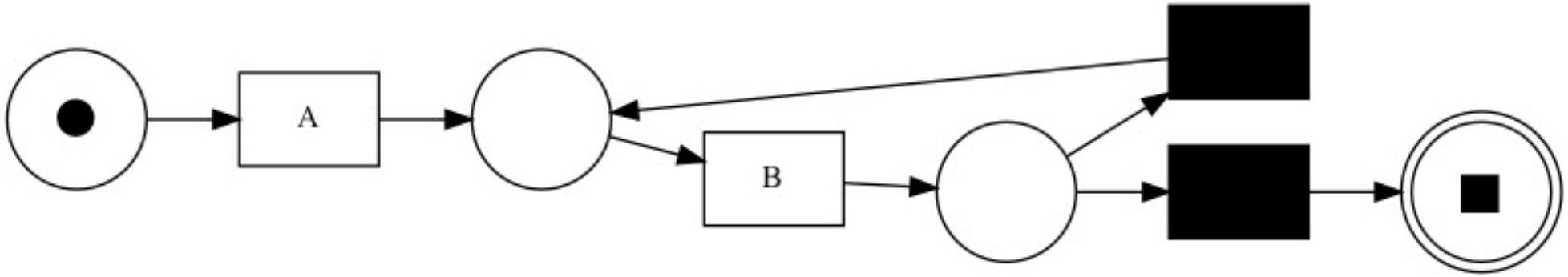
$$\mathit{Initial} = [\mathrm{A}],\ \mathit{final} = [\mathrm{B}]$$

$$\mathrm{L} = [< \mathit{LPM}{:}1\_0, \mathit{LPM}{:}1\_0, c, \mathit{LPM}{:}1\_0 >]$$

Eliminazione del LPM: 1

Esempio 4

$$\mathrm{L} = [< a(1), b(1), a(1), b(1), c, a(1), b(1) >]$$



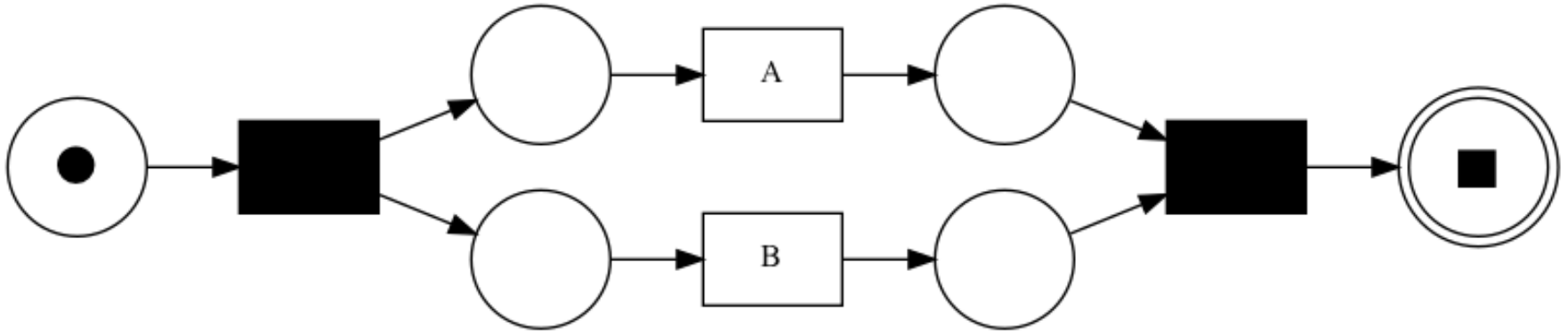
$$\mathit{Initial} = [\mathrm{A}],\ \mathit{final} = [\mathrm{B}]$$

$$\mathrm{L} = [< \mathit{LPM}\text{:}1\_0, \mathit{LPM}\text{:}1\_0, c, \mathit{LPM}\text{:}1\_0 >]$$

Eliminazione del LPM: 1

Esempio 5

$$\mathrm{L} = [< a(1), b(1), c, a(1), b(1), a >]$$



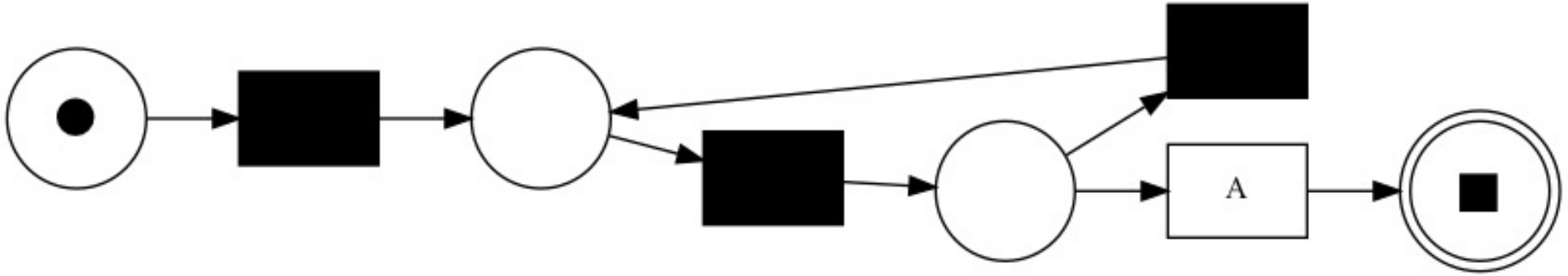
$$\mathit{Initial} = [\mathrm{A,B}],\ \mathit{final} = [\mathrm{A,B}]$$

$$\mathrm{L} = [< \mathit{LPM}\mathrm{:1\_0}, c, \mathit{LPM}\mathrm{:1\_0}, a >]$$

Eliminazione del LPM: 1

Esempio 6

$$\mathrm{L} = [< a(1), c, a(1), a(1) >]$$



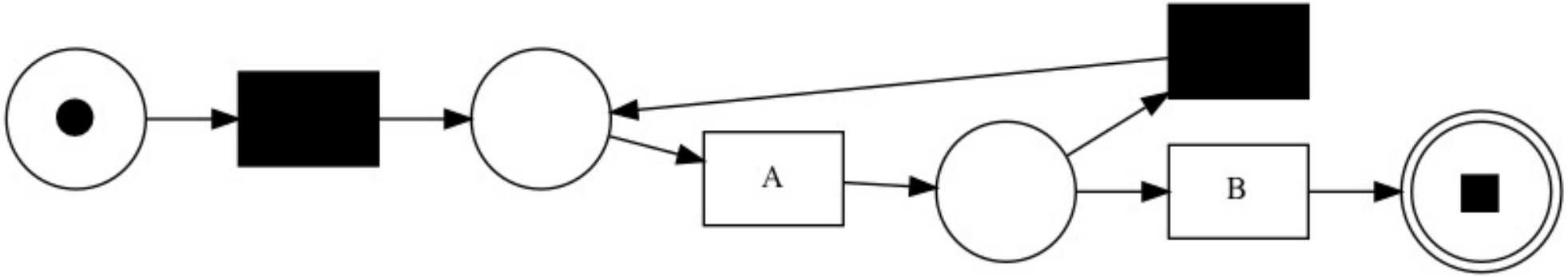
$$\mathit{Initial} = [\mathrm{A}],\ \mathit{final} = [\mathrm{A}]$$

$$\mathrm{L} = [< \mathit{LPM}\text{:}1\_0, c, \mathit{LPM}\text{:}1\_0, \mathit{LPM}\text{:}1\_0 >]$$

Eliminazione del LPM: 1

Esempio 7

$$\mathrm{L} = [< a(1), a(1), a(1), b(1), c, a(1), b(1) >]$$



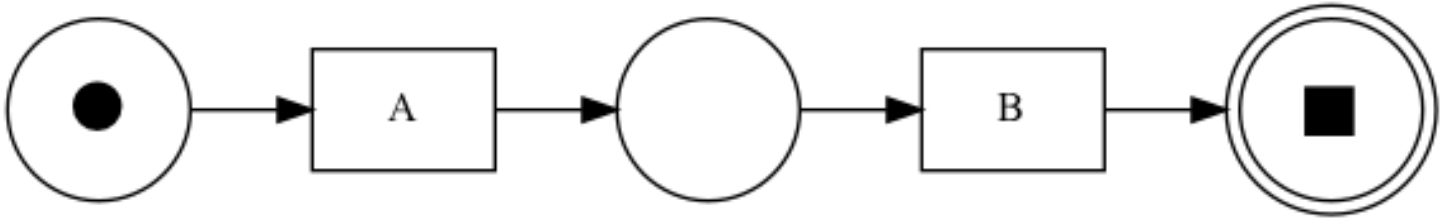
$$\mathit{Initial} = [\mathrm{A}],\ \mathit{final} = [\mathrm{B}]$$

$$\mathrm{L} = [< \mathit{LPM}: 1\_0, c, \mathit{LPM}: 1\_0 >]$$

Eliminazione del LPM: 1

Esempio 8

$$\mathrm{L} = [< a(1), c, b(1) >]$$



$$\mathit{Initial} = [\mathrm{A}], \mathit{final} = [\mathrm{B}]$$

$$\mathrm{L} = [< \mathit{LPM}{:}1\_0, c >]$$

Conclusioni e sviluppi futuri

L'obiettivo del seguente progetto è di realizzare un algoritmo di compressione che operi con la stessa logica di ***Subdue***, accettando in input *file di log* anziché grafi.

Alcuni sviluppi futuri potrebbero coinvolgere:

- Ottimizzazione dell'algoritmo di etichettatura;
- Realizzazione di un programma che inglobi le fasi di generazione dei LPM (al momento realizzato tramite interfaccia grafica da *ProM*) ed il qui proposto algoritmo di compressione.