Probabilistic Logic [Programming]

Outline:

1. **Κάποιες Θεωρητικές Προσεγγίσεις γύρω απο το θέμα**
2. **Case Study της προσέγγισης μας**
3. **Σύντομο Intro στην ProbLog.**

Probabilistic Logic [Programming]

Πρώτο Paper, γεννιέται ο όρος Probabilistic Logic
Ορίζει τα “Truth Values” με πιθανότητες.
Nilsson, Nils J. "Probabilistic logic." *Artificial intelligence* 28.1 (1986): 71-87.

Τα Λογικά Προγράμματα είναι ιδανικά για Rule-Based συστήματα.
Όμως τι γίνεται όταν οι πιθανότητες εμπλέκονται στο παιχνίδι;

Μας δίνουν την δυνατότητα να περιγράψουμε καταστάσεις
που περιλαμβάνουν
αβεβαιότητα.

Σχέση με Fuzzy Logic

Σε Fuzzy Λογικές έχουμε την έννοια της μερικής αλήθειας, ενώ στο Probabilistic Logic έχουμε την πιθανότητα να ισχύει κάτι.

Με το ένα Πόδι μέσα με το άλλο έξω…

Markov Logic Networks & ProbLog

Η Prolog είναι μια γλώσσα ικανή να αναπαραστήσει First-Order Logic.

Η ProbLog προσθέτει πιθανότητες στις προτάσεις της prolog.

Markov Logic Networks & ProbLog

Η Prolog είναι μια γλώσσα ικανή να αναπαραστήσει First-Order Logic.

Η ProbLog προσθέτει πιθανότητες στις προτάσεις της prolog.

Με την ProbLog μπορούμε αν περιγράψουμε ένα δίκτυο Markov Random Field μαζί με άλλα facts.

Markov Logic Networks & ProbLog

Η Prolog είναι μια γλώσσα ικανή να αναπαραστήσει First-Order Logic.

Η ProbLog προσθέτει πιθανότητες στις προτάσεις της prolog.

Με την ProbLog μπορούμε αν περιγράψουμε ένα δίκτυο Markov Random Field μαζί με άλλα facts.

Τα Markov Random Fields κάνουν ισχυρές υποθέσεις ανεξαρτησίας οι οποίες μας επιτρέπουν να κάνουμε inference.

Markov Logic Networks & ProbLog

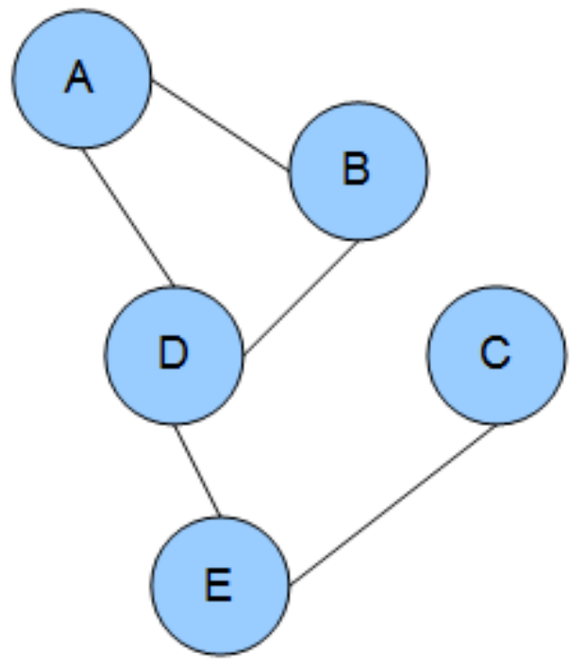
Η ProbLog πρακτικά μας δίνει έναν τρόπο να λειτουργούμε πάνω σε Markov Logic Networks.

Μας επιτρέπει να κάνουμε Inference και να υπολογίσουμε Marginals

$$\Pr(X = x) = \sum_y \Pr(X = x, Y = y) = \sum_y \Pr(X = x \mid Y = y) \Pr(Y = y),$$

ProbLog [example]

```
1   0.3::stress(X) :- person(X).
2   0.2::influences(X,Y) :- person(X), person(Y).
3
4   smokes(X) :- stress(X).
5   smokes(X) :- friend(X,Y), influences(Y,X), smokes(Y).
6
7   0.4::asthma(X) :- smokes(X).
8
9   person(1).
10  person(2).
11  person(3).
12  person(4).
13
14  friend(1,2).
15  friend(2,1).
16  friend(2,4).
17  friend(3,2).
18  friend(4,2).
19
```



ProbLog [example]

```
20  evidence(smokes(2),true).
21  evidence(influences(4,2),false).
22
23  query(smokes(1)).
24  query(smokes(3)).
25  query(smokes(4)).
26  query(asthma(1)).
27  query(asthma(2)).
28  query(asthma(3)).
29  query(asthma(4)).
```

ProbLog [example]

```
1  0.3::stress(X) :- person(X).
2  0.2::influences(X,Y) :- person(X), person(Y).
3
4  smokes(X) :- stress(X).
5  smokes(X) :- friend(X,Y), influences(Y,X), smokes(Y).
6
7  0.4::asthma(X) :- smokes(X).
8
9  person(1).
10 person(2).
11 person(3).
12 person(4).
13
14 friend(1,2).
15 friend(2,1).
16 friend(2,4).
17 friend(3,2).
18 friend(4,2).
19
```

```
20 evidence(smokes(2),true).
21 evidence(influences(4,2),false).
22
23 query(smokes(1)).
24 query(smokes(3)).
25 query(smokes(4)).
26 query(asthma(1)).
27 query(asthma(2)).
28 query(asthma(3)).
29 query(asthma(4)).
```

Query▼	Location	Probability
asthma(1)	26:7	0.20350877
asthma(2)	27:7	0.4
asthma(3)	28:7	0.176
asthma(4)	29:7	0.176
smokes(1)	23:7	0.50877193
smokes(3)	24:7	0.44
smokes(4)	25:7	0.44

Decision Theoretic Probabilistic Prolog

**“the utility of a strategy (a particular choice of actions)
is defined as the expected reward for its execution in the presence
of probabilistic effects”**

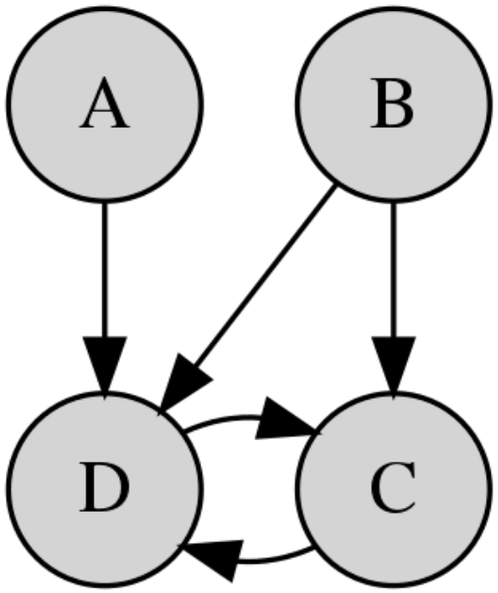
a set of decision facts, specifying which decisions are to be made

and a set of utility attributes, specifying the rewards that can be obtained.

Guy Van den Broeck and Ingo Thon and Martijn van Otterlo and Luc De Raedt
DTPROBLOG: A Decision-Theoretic Probabilistic Prolog

Probabilistic Graphical Models

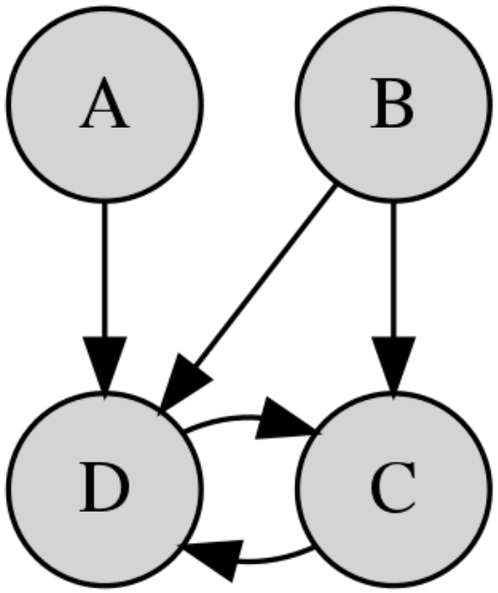
Probabilistic Μοντέλο με την χρήση γράφου, ο οποιός περιγράφει την εξάρτηση μεταξύ τυχαίων μεταβλητών.



- **Ο D εξαρτάται απο τον A και τον B**
- **Ο D και ο C αλληλοεξαρτώνται**
- **Ο B δεν εξαρτάται απο κάποιον.**

Probabilistic Graphical Models

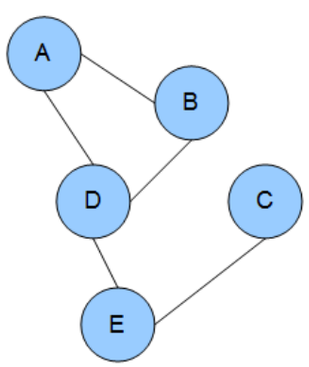
Probabilistic Μοντέλο με την χρήση γράφου, ο οποιός περιγράφει την εξάρτηση μεταξύ τυχαίων μεταβλητών.



- **Ο D εξαρτάται απο τον A και τον B**
- **Ο D και ο C αλληλοεξαρτώνται**
- **Ο B δεν εξαρτάται απο κάποιον.**

Bayesian Networks: DAG

Markov Random Field:
Undirected, Maybe Acyclic



wikipedia: source of images

Probabilistic Graphical Models

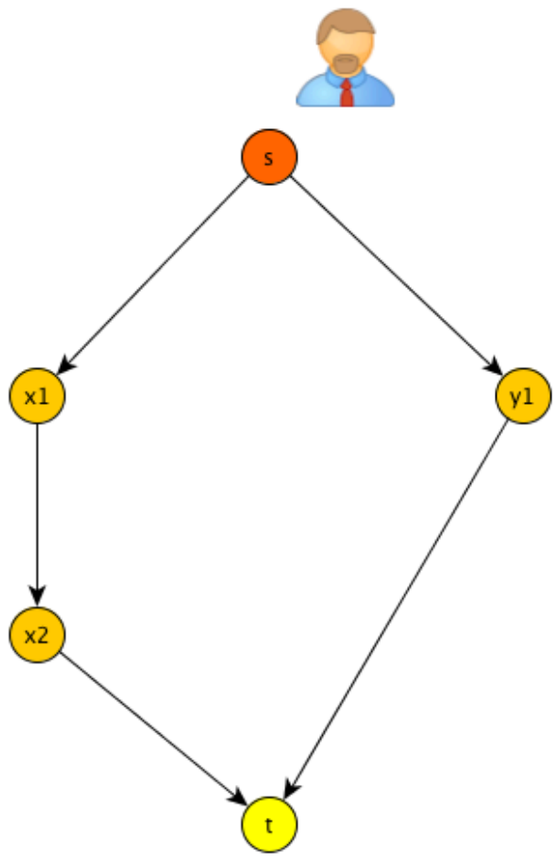
Στο **Statistical Relational Learning (SRL)** και στα **Graphical Models** το ζητούμενο είναι να βρούμε την **marginal** πιθανότητα, δεδομένων κάποιων **evidence (MARG)**

Θα θέλαμε επίσης την πιο πιθανή κοινή κατάσταση των μεταβλητών δεδομένων των **evidence (MPE).**

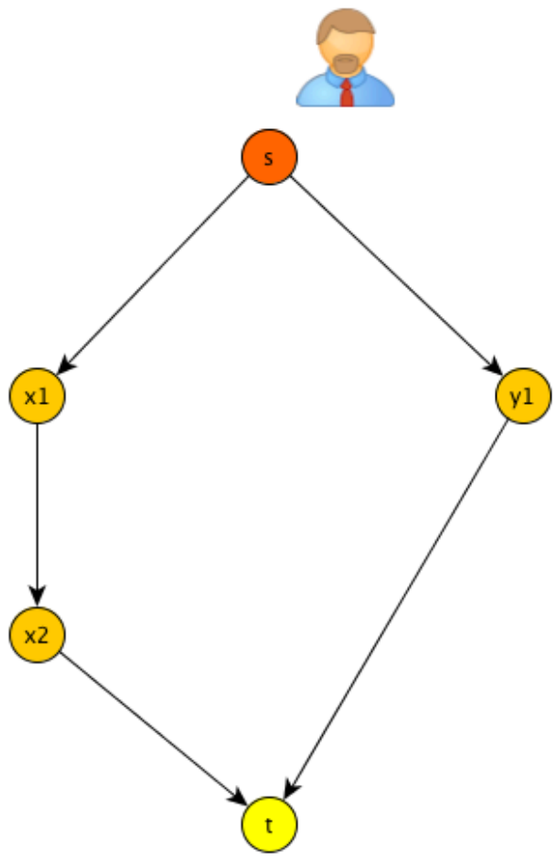
Example Case: Simple Network

Example 1: Simple Network

Θέλω να φτάσω απο το s στο t.



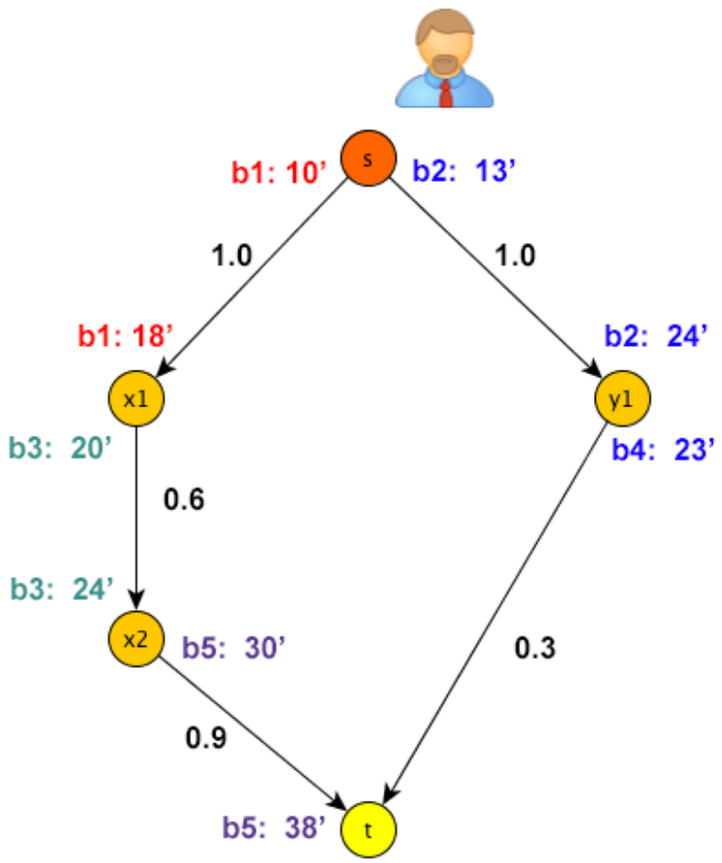
Example 1: Simple Network



Θέλω να φτάσω απο το s στο t.

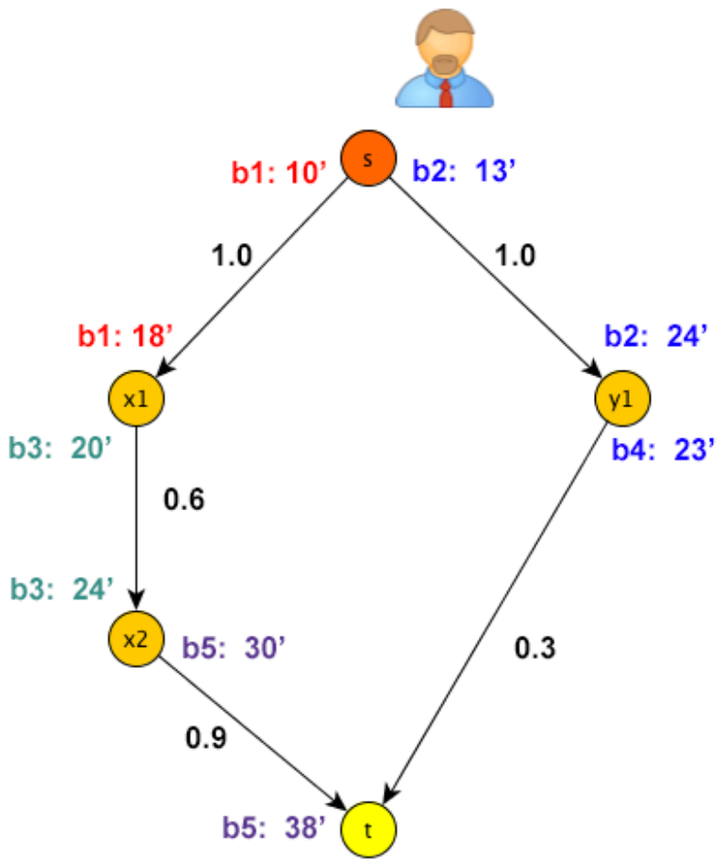
Κάθε Κόμβος συμβολίζει μια στάση στην οποία πρέπει να κάνω αλλαγή λεωφορείου

Example 1: Simple Network



Δεδομένα: **Η εκτίμηση για το πότε θα έρθει το επόμενο λεωφορείο. Στην περίπτωση του OASA είναι δεδομένη. Μπορεί να υπολογίζεται με βάση του τι έχει συμβεί στο παρελθόν.**

Example 1: Simple Network



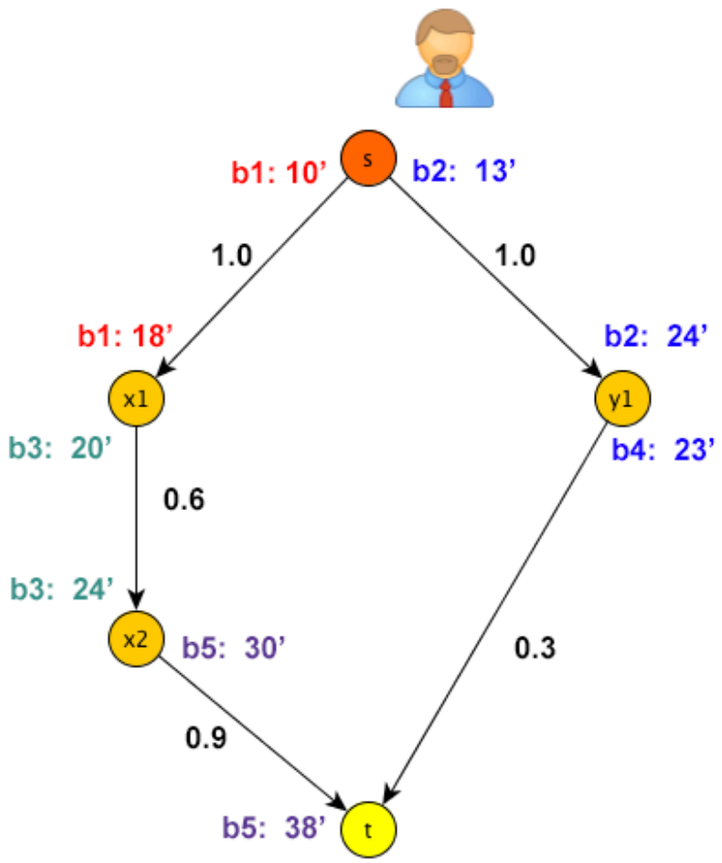
Δεδομένα: **Η εκτίμηση για το πότε θα έρθει το επόμενο λεωφορείο. Στην περίπτωση του OASA είναι δεδομένη. Μπορεί να υπολογίζεται με βάση του τι έχει συμβεί στο παρελθόν.**

ΜΠΟΡΟΥΜΕ ΝΑ ΤΑ ΥΠΟΛΟΓΙΣΟΥΜΕ ΚΑΙ ΕΜΕΙΣ!

Maximum Likelihood μας δίνει, με την υπόθεση του μοντέλου μας:

$$\hat{\mu}_{\mathrm{ML}} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \hat{\Sigma}_{\mathrm{ML}} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu}_{\mathrm{ML}})(x_i - \hat{\mu}_{\mathrm{ML}})^T.$$

Example 1: Simple Network



Δεδομένα: **Η εκτίμηση για το πότε θα έρθει το επόμενο λεωφορείο. Στην περίπτωση του OASA είναι δεδομένη. Μπορεί να υπολογίζεται με βάση του τι έχει συμβεί στο παρελθόν.**

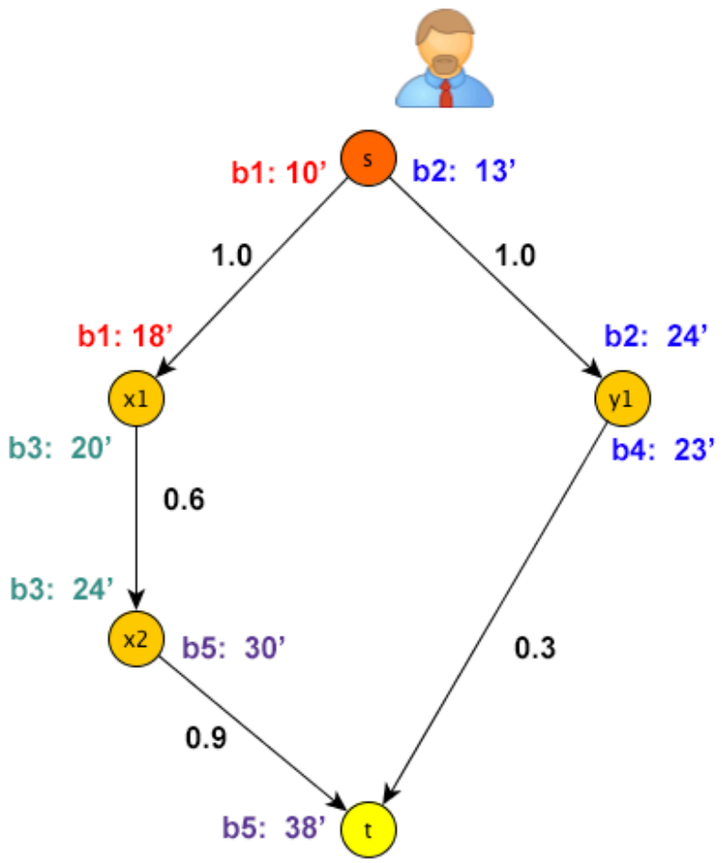
ΜΠΟΡΟΥΜΕ ΝΑ ΤΑ ΥΠΟΛΟΓΙΣΟΥΜΕ ΚΑΙ ΕΜΕΙΣ!

Maximum Likelihood μας δίνει, με την υπόθεση του μοντέλου μας:

$$\hat{\mu}_{\mathrm{ML}} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \hat{\Sigma}_{\mathrm{ML}} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu}_{\mathrm{ML}})(x_i - \hat{\mu}_{\mathrm{ML}})^T.$$

Δηλαδή απλός υπολογισμός με βάση προηγούμενες μετρήσεις.

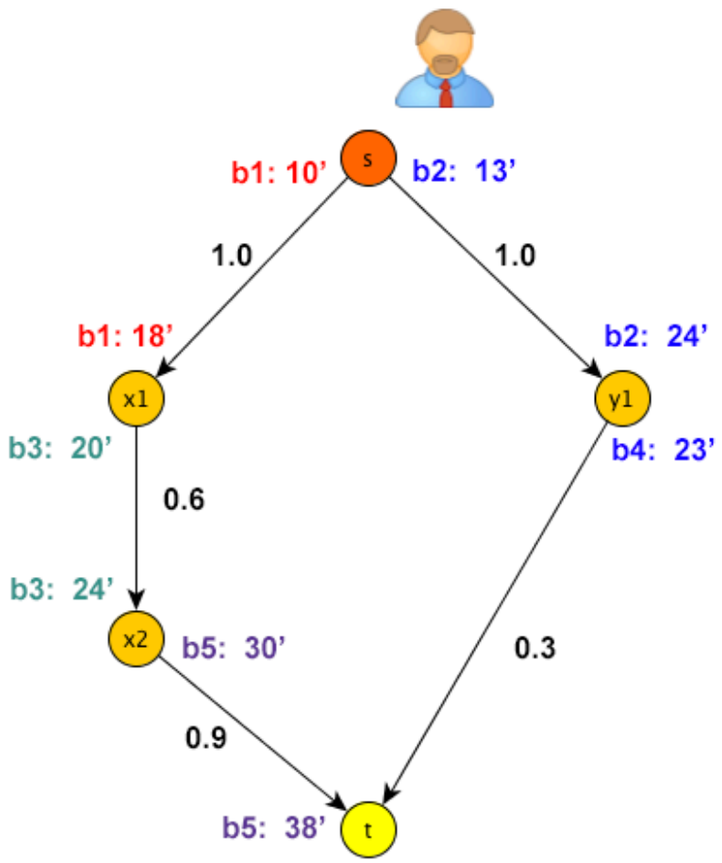
Example 1: Simple Network



Δεδομένα: **Η εκτίμηση για το πότε θα έρθει το επόμενο λεωφορείο. Στην περίπτωση του OASA είναι δεδομένη. Μπορεί να υπολογίζεται με βάση του τι έχει συμβεί στο παρελθόν.**

Πρόβλημα: Αν πρόκειται να κάνουμε αλλαγές λεωφορείων, είναι δύσκολο να διαχειριστούμε όλη αυτή την πληροφορία.. Πόσο μάλλον δεδομένου ότι αναφερόμαστε σε εκτιμήσεις

Example 1: Simple Network



Δεδομένα: **Η εκτίμηση για το πότε θα έρθει το επόμενο λεωφορείο. Στην περίπτωση του OASA είναι δεδομένη. Μπορεί να υπολογίζεται με βάση του τι έχει συμβεί στο παρελθόν.**

Πρόβλημα: Αν πρόκειται να κάνουμε αλλαγές λεωφορείων, είναι δύσκολο να διαχειριστούμε όλη αυτή την πληροφορία.. Πόσο μάλλον δεδομένου ότι αναφερόμαστε σε εκτιμήσεις

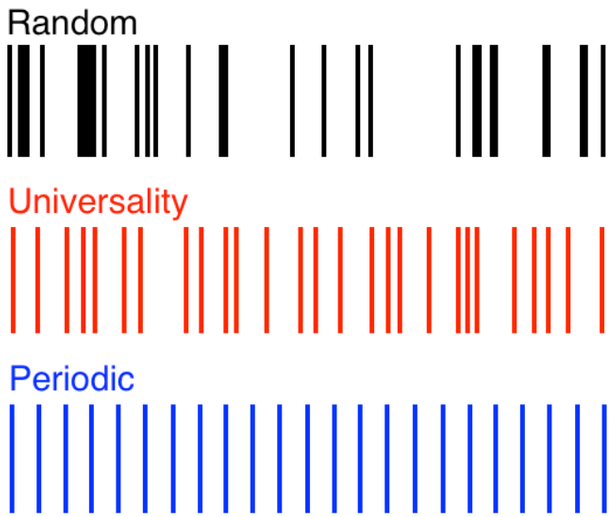
Όπως καταλαβαίνουμε ο χρόνος άφιξης μπορεί να μοντελοποιηθεί σαν μια τυχαία μεταβλητή…

Low - Level παράμετροι που επηρεάζουν την τυχαία μεταβλητή

- Κίνηση/Μποτιλιάρισμα/Τρακαρίσματα
- Φανάρια (Δημιουργείται κάποια ουρά αναμονής)

High - Level Μοντελοποίηση

- Πραγματικότητα πολύ χαοτική (Φαινόμενο **Universality**)



Κανονικότητα μέσα στην τυχαιότητα των λεωφορείων, Cuernavaca, Mexico

- Μεγάλο Υπολογιστικό κόστος, σύνθετο πρόβλημα.

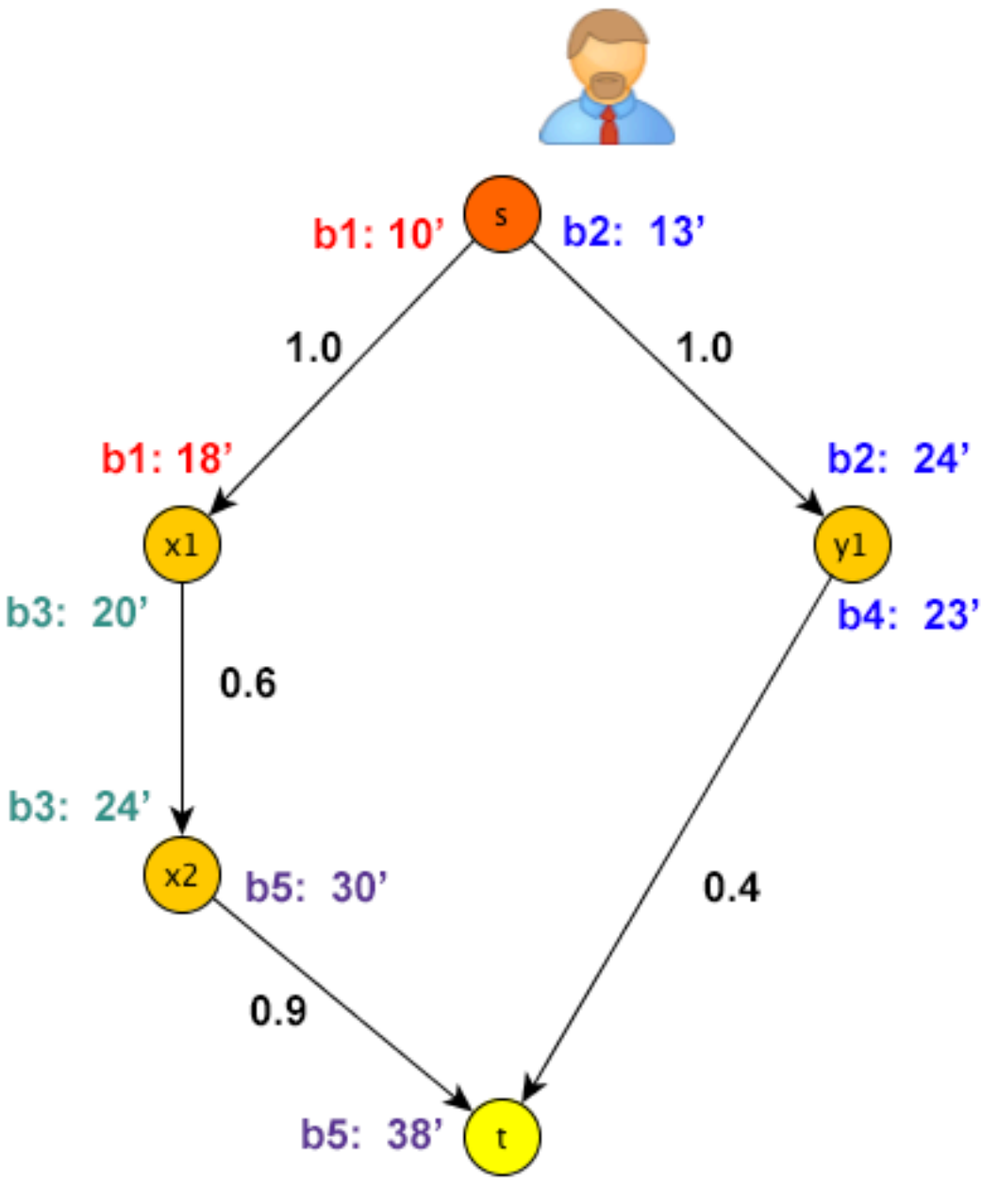
- Μοντελοποιούμε τους χρόνους αναμονής σαν κανονική κατανομή.

High - Level Μοντελοποίηση

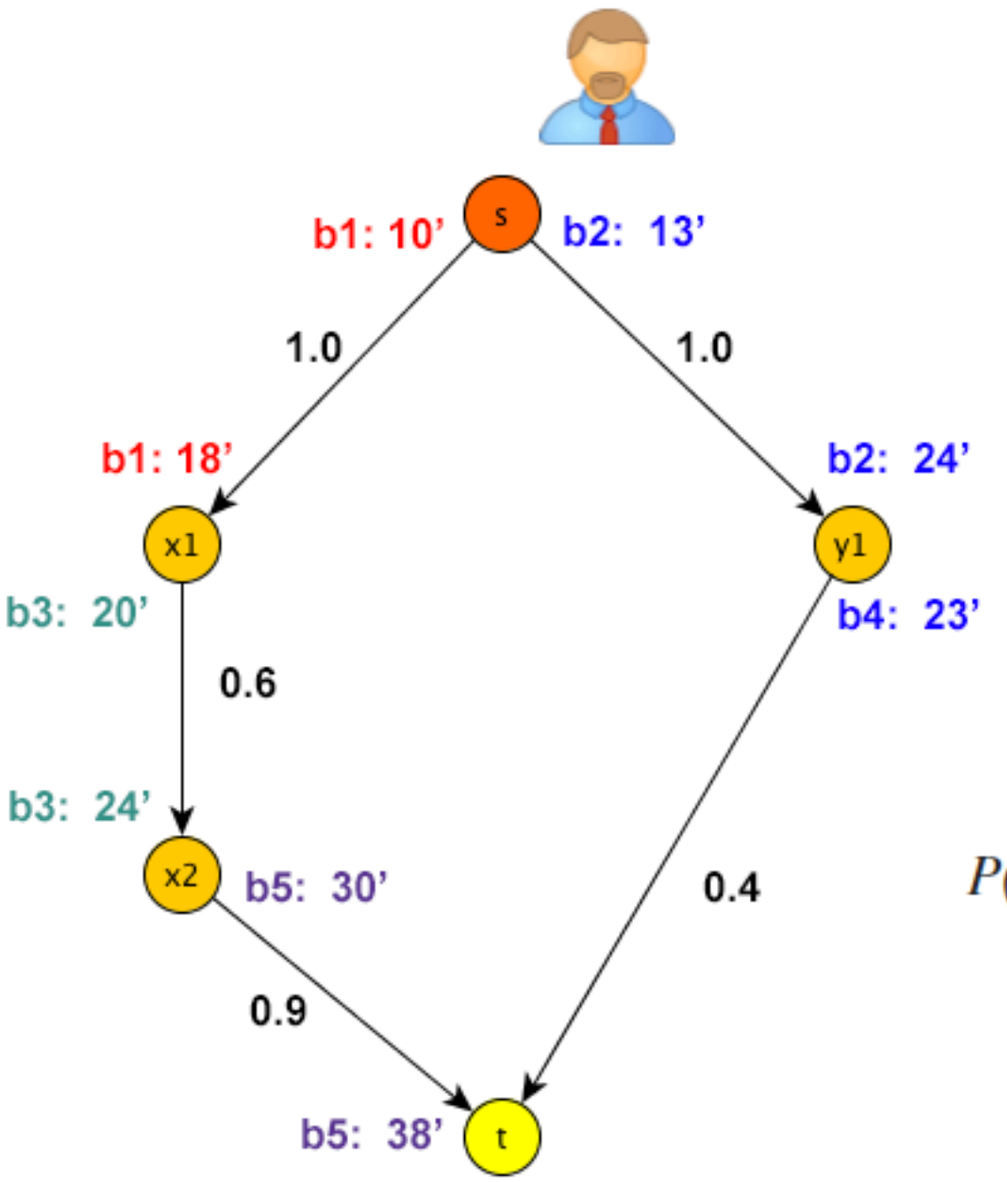
- Η τυπική απόκλιση θα πρέπει να προκύπτει απο παρελθοντικές τιμές.

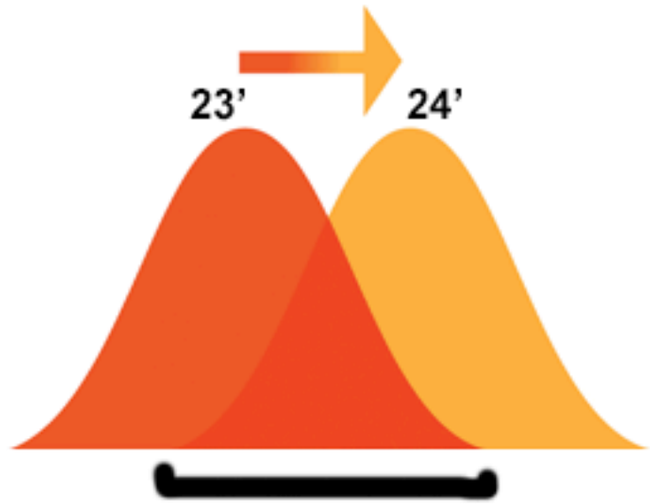
- Σε ένα μοντέλο “**εντός πόλης**”, μπορεί να χρησιμοποιηθεί τιμή ανάλογη της εκτίμησης για την άφιξη στον προορισμό.

Example 1: Simple Network



Example 1: Simple Network





$$P(X_1>X_2)=P(X_1-X_2>0)=1-P(X_1-X_2\leq 0).$$

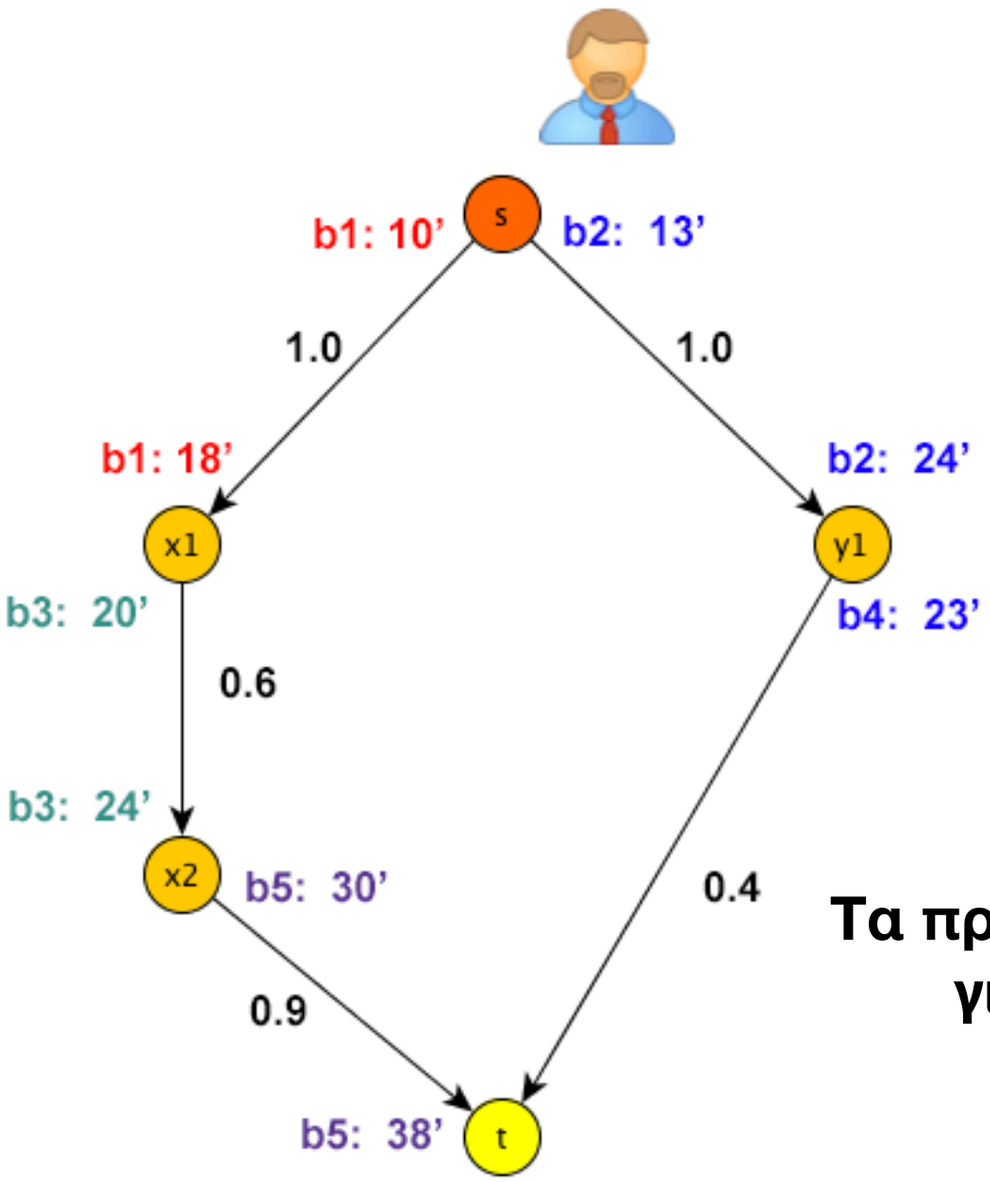
(Υπόθεση: Ανεξαρτησία)

$$\mu:=E(X_1-X_2)=\mu_1-\mu_2$$

$$\sigma^2:=Var(X_1-X_2)=\sigma_1^2+\sigma_2^2.$$

$$P(X_1>X_2)=1-P(X_1-X_2\leq 0)=1-\Phi\left(\frac{-\mu}{\sigma}\right).$$

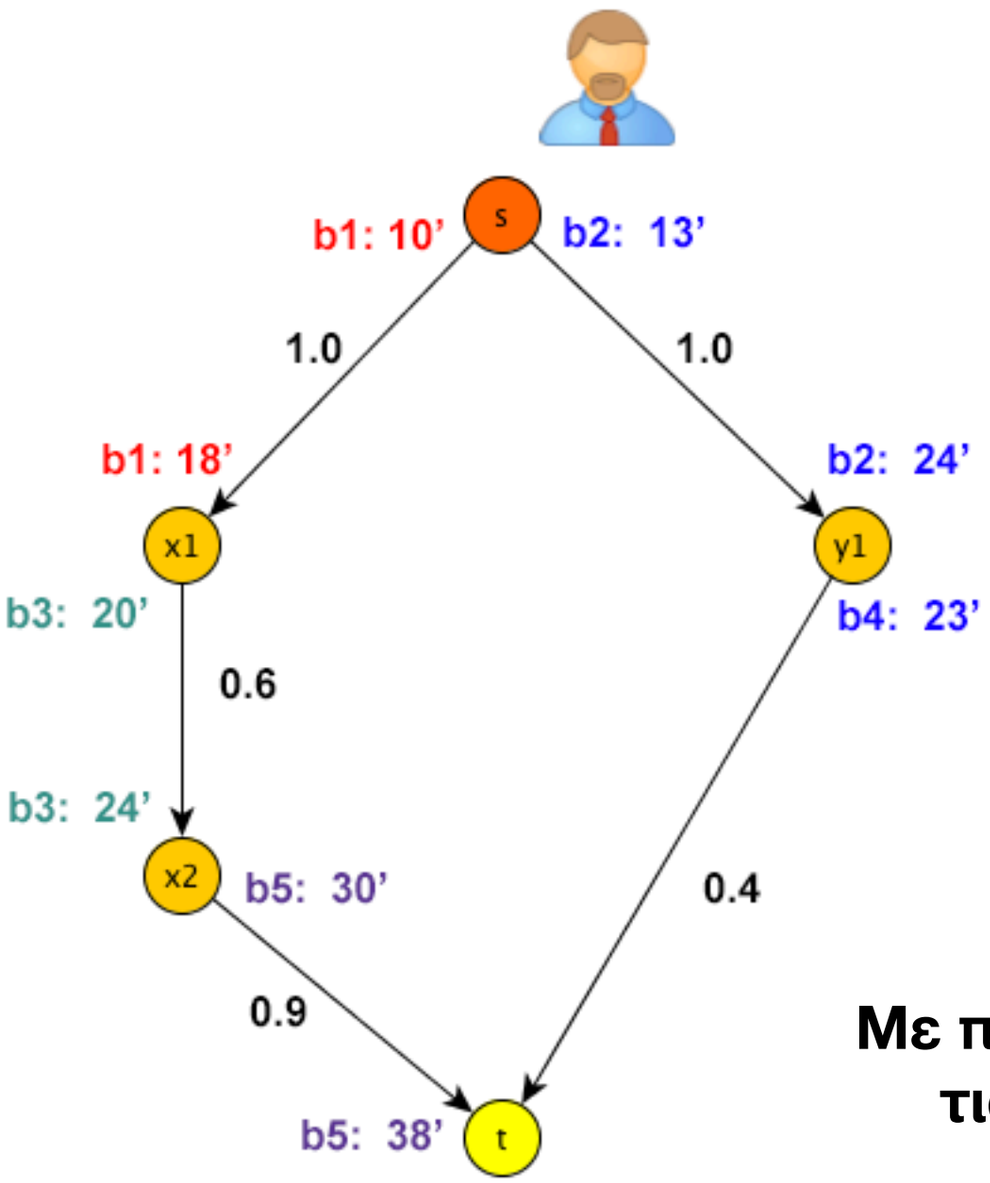
Example 1: Simple Network



Τα πράγματα γίνονται αρκετά περίπλοκα για αναλυτικές λύσεις αν έχουμε

$$P(X_1>X_2>X_3>X_4>\ldots)$$

Example 1: Simple Network

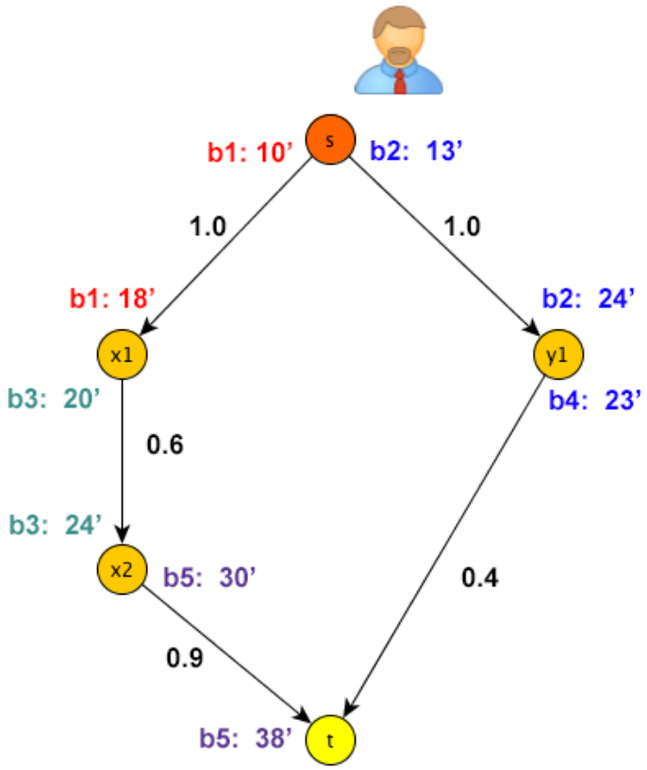


Προσομοίωση!

Με προσομοίωση θα υπολογίσουμε τις πιθανότητες κάθε κόμβου.

Έπειτα θα χρησιμοποιήσουμε Probabilistic Logic Programming για την μελέτη των αποτελεσμάτων

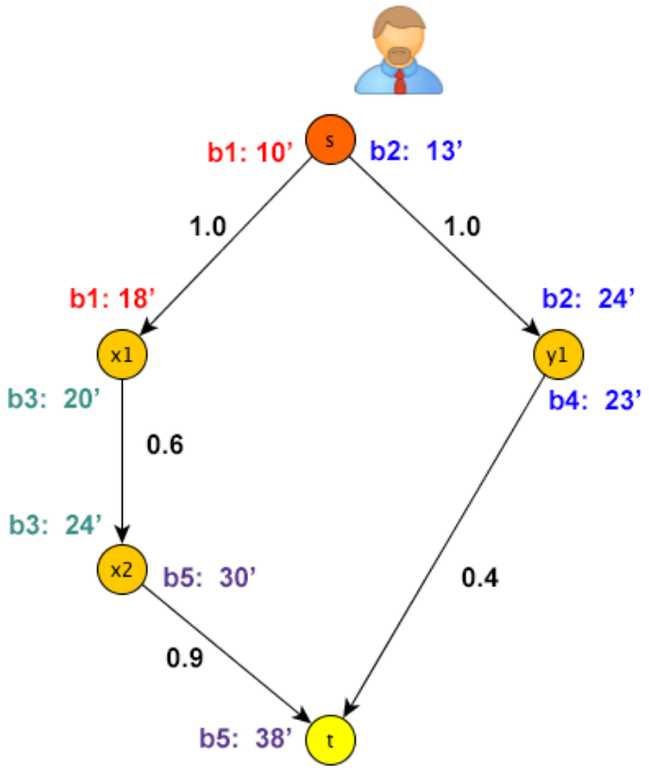
Example 1: Simple Network



Αποτελέσματα προσομοίωσης με την υπόθεση: ότι τυπική απόκλιση: estimation/10

1.0000	1.0000
0.7563	0.3897
0.9026	

Example 1: Simple Network




```
1   edge(s,xf).
2   0.7563 ::edge(xf,xs).
3   0.9026 ::edge(xs,t).
4
5   edge(s,yf).
6   0.3897 ::edge(yf,t).
7
8   path(X,Y) :- edge(X,Y).
9   path(X,Y) :- edge(X,Z),
10                Y \== Z,
11           path(Z,Y).
12
13  query(path(s,t)).
```

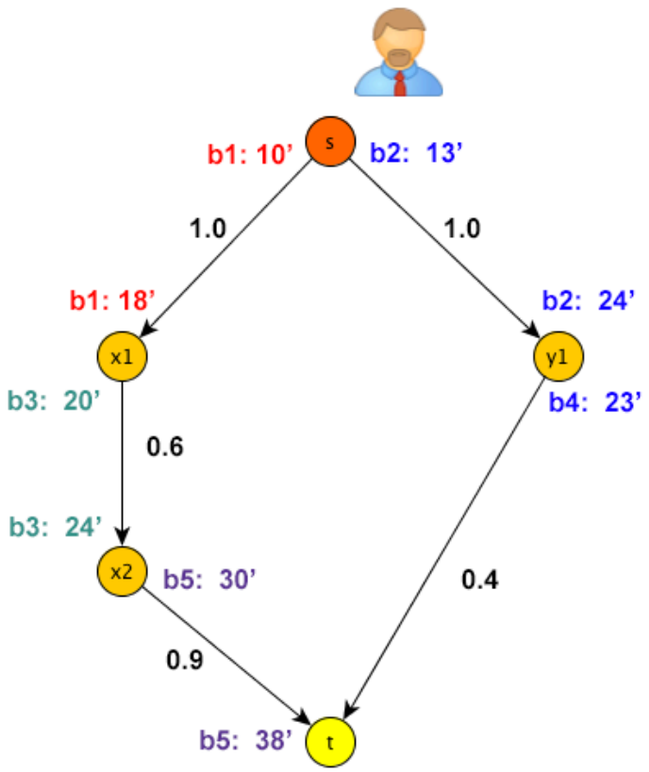
1.0000
0.7563
0.9026

1.0000
0.3897

Πιθανότητα να φτάσω στον προορισμό μου με χρήση κάποιου μονοπατιού

0.80631298

Example 1: Simple Network




```
1   %edge(s,xf).
2   0.7563 ::edge(xf,xs).
3   0.9026 ::edge(xs,t).
4
5   edge(s,yf).
6   0.3897 ::edge(yf,t).
7
8   path(X,Y) :- edge(X,Y).
9   path(X,Y) :- edge(X,Z),
10                Y \== Z,
11           path(Z,Y).
12
13  query(path(s,t)).
```

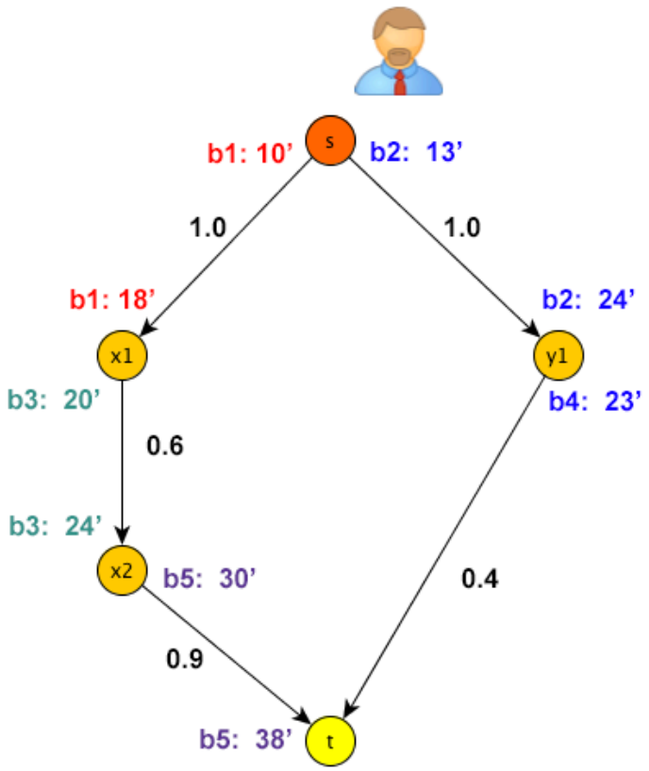
1.0000
0.7563
0.9026

1.0000
0.3897

Πιθανότητα να φτάσω στον προορισμό μου απο τα δεξιά

0.3897

Example 1: Simple Network




```
1   edge(s,xf).
2   0.7563 ::edge(xf,xs).
3   0.9026 ::edge(xs,t).
4
5   %edge(s,yf).
6   0.3897 ::edge(yf,t).
7
8   path(X,Y) :- edge(X,Y).
9   path(X,Y) :- edge(X,Z),
10                Y \== Z,
11           path(Z,Y).
12
13  query(path(s,t)).
```

1.0000
0.7563
0.9026

1.0000
0.3897

Πιθανότητα να φτάσω στον προορισμό μου απο τα αριστερά

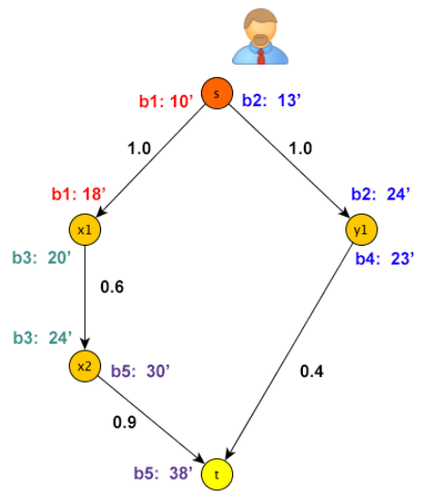
0.68263638

Σε πιο περίπλοκα δίκτυα σημαντική διευκόλυνση!

Example 1: Simple Network

Με την Decision Theoretic Problog μπορώ να αναθέσω την επιλογή του μονοπατιού στο πρόγραμμα!

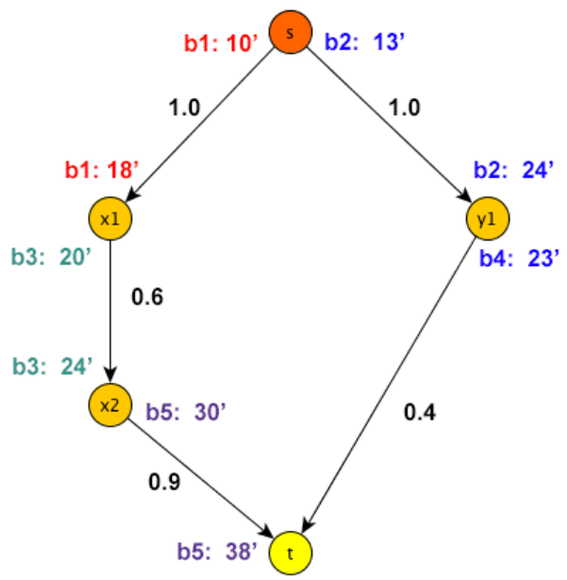
```
1   ?::left_path.
2   ?::right_path.
3
4   both_taken :- left_path, right_path.
5
6   edge(s,x1) :- left_path.
7   0.7563 ::edge(x1,x2).
8   0.9026 ::edge(x2,t).
9
10  edge(s,y1) :- right_path.
11  0.3897 ::edge(y1,t).
12
13  path(X,Y) :- edge(X,Y).
14  path(X,Y) :- edge(X,Z),
15               Y \== Z,
16           path(Z,Y).
17
18  utility(left_path , -38). % Minutes in the Left Path
19  utility(right_path, -36). % Minutes in the Right Path
20  utility(both_taken, -100000).
21  utility(path(s,t), 100).
```



Example 1: Simple Network

Με την Decision Theoretic Problog μπορώ να αναθέσω την επιλογή του μονοπατιού στο πρόγραμμα!

```
1   ?::left_path.
2   ?::right_path.
3
4   both_taken :- left_path, right_path.
5
6   edge(s,x1) :- left_path.
7   0.7563 ::edge(x1,x2).
8   0.9026 ::edge(x2,t).
9
10  edge(s,y1) :- right_path.
11  0.3897 ::edge(y1,t).
12
13  path(X,Y) :- edge(X,Y).
14  path(X,Y) :- edge(X,Z),
15               Y \== Z,
16           path(Z,Y).
17
18  utility(left_path , -38). % Minutes in the Left Path
19  utility(right_path, -36). % Minutes in the Right Path
20  utility(both_taken, -100000).
21  utility(path(s,t), 100).
```



Example 1: Simple Network

```
1   ?::left_path.
2   ?::right_path.
3
4   both_taken :- left_path, right_path.
5
6   edge(s,x1) :- left_path.
7   0.7563 ::edge(x1,x2).
8   0.9026 ::edge(x2,t).
9
10  edge(s,y1) :- right_path.
11  0.3897 ::edge(y1,t).
12
13  path(X,Y) :- edge(X,Y).
14  path(X,Y) :- edge(X,Z),
15               Y \== Z,
16           path(Z,Y).
17
18  utility(left_path , -38). % Minutes in the Left Path
19  utility(right_path, -36). % Minutes in the Right Path
20  utility(both_taken, -100000).
21  utility(path(s,t), 100).
```

Πρέπει να επιλέξουμε κατάλληλα παραμέτρους και την σημασία που έχουν για εμάς

Example 1: Simple Network

```
1   ?::left_path.
2   ?::right_path.
3
4   both_taken :- left_path, right_path.
5
6   edge(s,x1) :- left_path.
7   0.7563 ::edge(x1,x2).
8   0.9026 ::edge(x2,t).
9
10  edge(s,y1) :- right_path.
11  0.3897 ::edge(y1,t).
12
13  path(X,Y) :- edge(X,Y).
14  path(X,Y) :- edge(X,Z),
15               Y \== Z,
16           path(Z,Y).
17
18  utility(left_path , -38). % Minutes in the Left Path
19  utility(right_path, -36). % Minutes in the Right Path
20  utility(both_taken, -100000).
21  utility(path(s,t), 100).
```

Πρέπει να επιλέξουμε κατάλληλα παραμέτρους και την σημασία που έχουν για εμάς

Ίσως το κόστος του χρόνου να αυξάνει για κάποιους τετραγωνικά!

ProbLog INTRO

Βασικό Παράδειγμα 1

```
% Γεγονότα
0.5::heads1.
0.6::heads2.

% Κανόνες
twoHeads :- heads1, heads2.

% Αιτήματα
query(heads1).
query(heads2).
query(twoHeads).
```

Βασικό Παράδειγμα 1

```
% Γεγονότα
0.5::heads1.
0.6::heads2.

% Κανόνες
twoHeads :- heads1, heads2.

% Αιτήματα
query(heads1).
query(heads2).
query(twoHeads).
```

ΠΙΘΑΝΟΤΗΤΑ ΝΑ ΣΥΜΒΟΥΝ ΚΑΙ ΤΑ ΔΥΟ

```
0.5
0.6
0.3
```

Βασικό Παράδειγμα 2

```
% Γεγονότα
0.5::heads1.
0.6::heads2.

% Κανόνες
someHeads :- heads1.
someHeads :- heads2.

% Αιτήματα
query(heads1).
query(heads2).
query(someHeads).
```

Βασικό Παράδειγμα 2

```
% Γεγονότα
0.5::heads1.
0.6::heads2.

% Κανόνες
someHeads :- heads1.
someHeads :- heads2.

% Αιτήματα
query(heads1).
query(heads2).
query(someHeads).
```

ΠΙΘΑΝΟΤΗΤΑ ΝΑ ΣΥΜΒΕΙ ΕΝΑ ΑΠΟ ΤΑ ΔΥΟ

0.5
0.6
0.8

First-Order Logic

```
% Γεγονότα
0.6::heads(C) :- coin(C).

% Δεδομένη Πληροφορία
coin(c1).
coin(c2).
coin(c3).
coin(c4).

% Κανόνες

someHeads :- heads(_).

% Αιτήματα
query(someHeads).
```

First-Order Logic

```
% Γεγονότα
0.6::heads(C) :- coin(C).

% Δεδομένη Πληροφορία
coin(c1).
coin(c2).
coin(c3).
coin(c4).

% Κανόνες

someHeads :- heads(_).

% Αιτήματα
query(someHeads).                0.9744 (4 ζάρια έρχονται heads)
```

$$P[Alarm|Burglary,Earthquake]=0.9$$

0.9::alarm :- burglary, earthquake

Inference?

```
0.7::burglary.
0.2::earthquake.

0.9::alarm :- burglary, earthquake.
0.8::alarm :- burglary, \+earthquake.
0.1::alarm :- \+burglary, earthquake.
```

```
evidence(alarm,true).
query(burglary).
query(earthquake).
```