KATHOLIEKE UNIVERSITEIT LEUVEN

DTAI
MACHINE LEARNING

DTProbLog

A Decision-Theoretic Probabilistic Prolog

Guy Van den Broeck
Ingo Thon
Martijn van Otterlo
Luc De Raedt

Motivation

- Many real-world decision problems are relational and probabilistic
  - Wildfire control
  - Who to vaccinate for swine flu
  - Viral marketing
  - ...

Motivation

- Combination of **relations**, **uncertainty** and **decision theory** largely ignored

  some exceptions are MLDNs, DTLPs and FOMDPs

- Relations with uncertainty

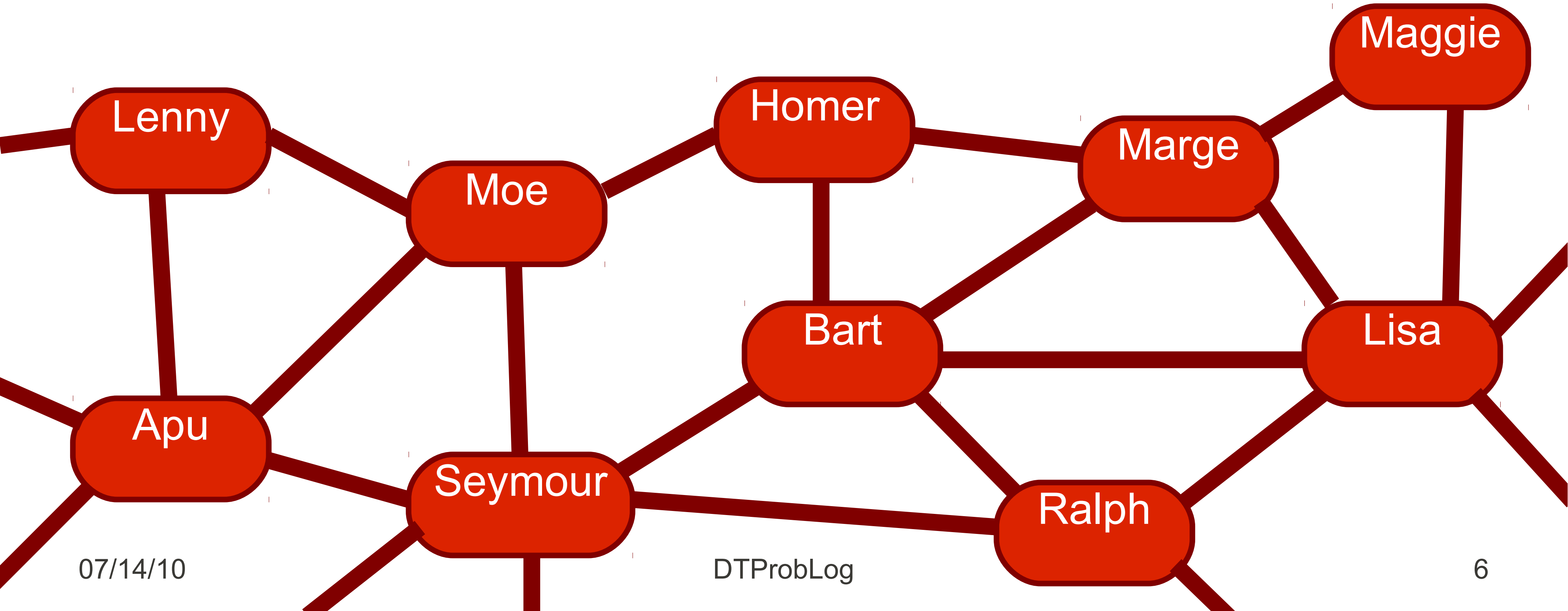
  = **S**tatistical **R**elational **L**earning

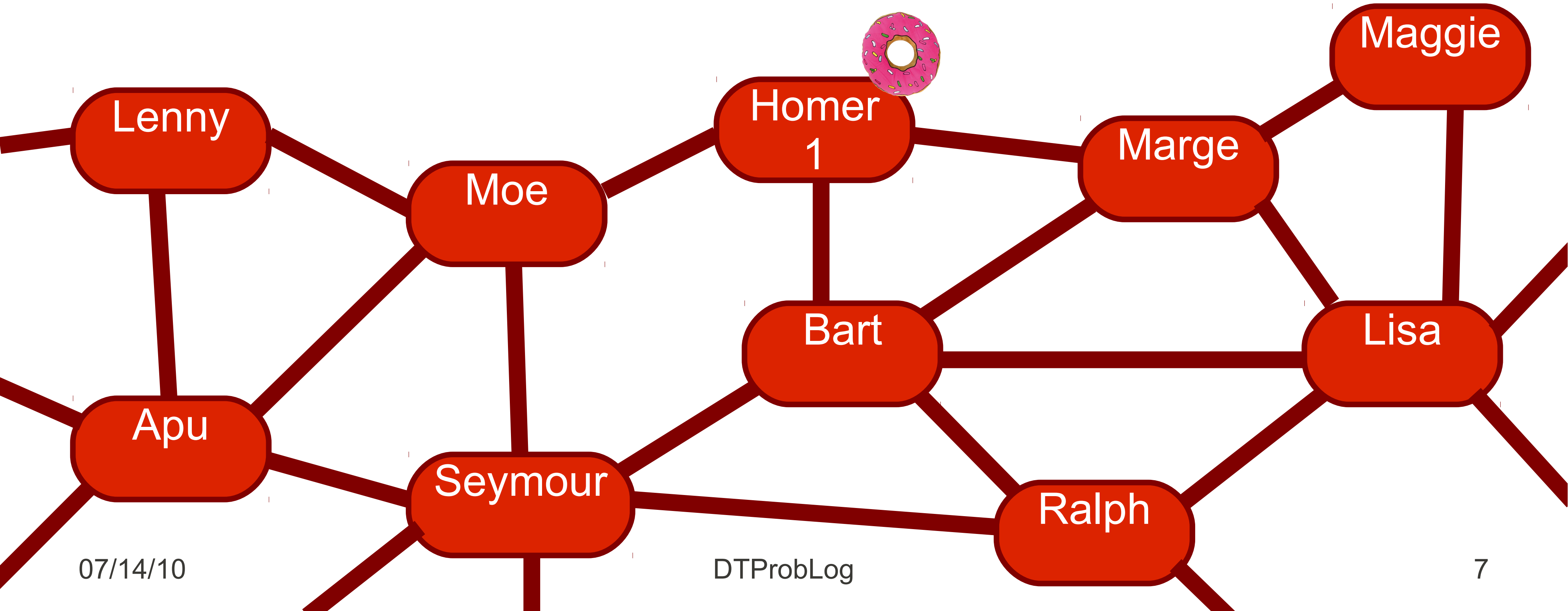
- ProbLog is a simple **probabilistic** Prolog
- DTProbLog is a **decision-theoretic** ProbLog

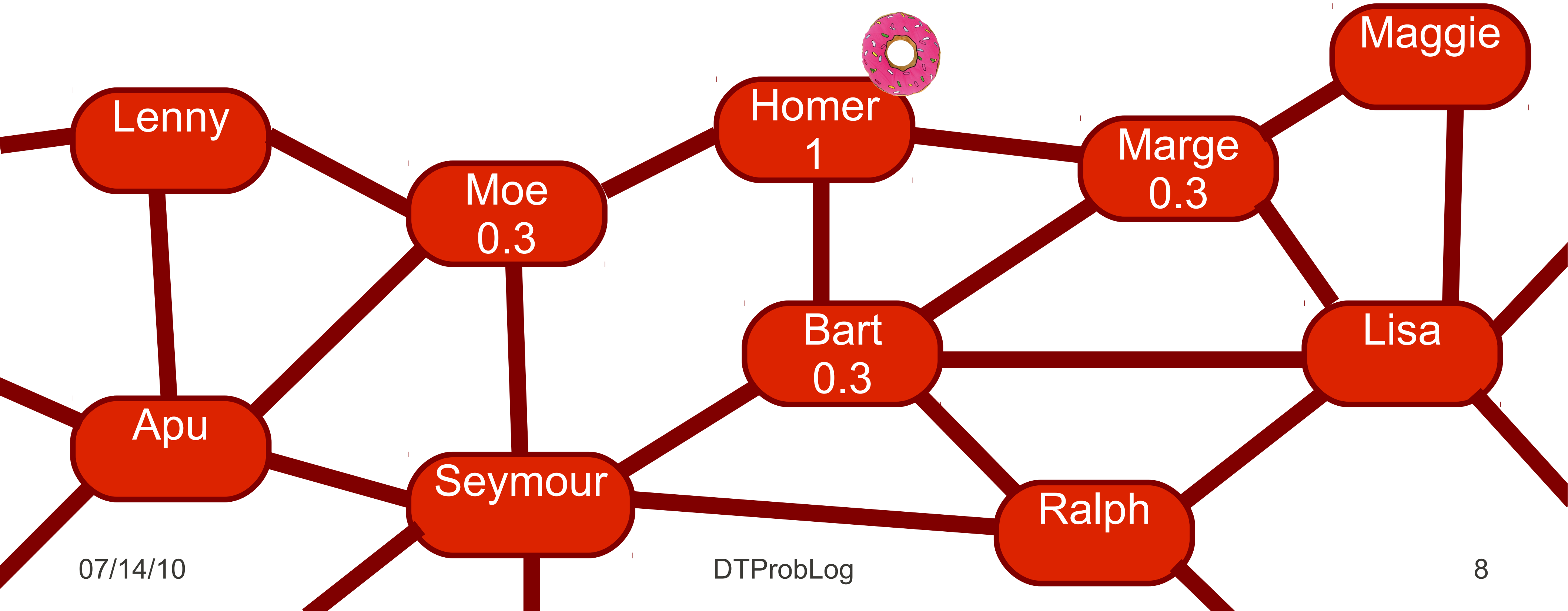
Outline

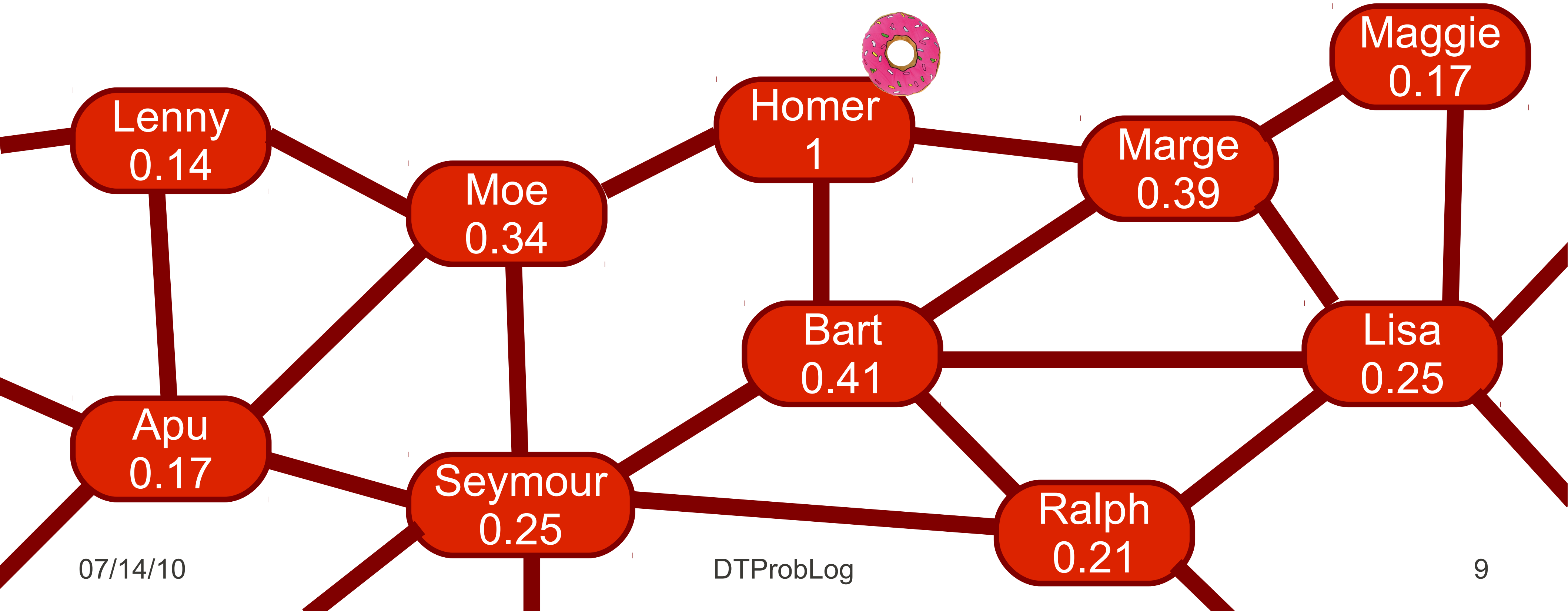
- DTProbLog: the Language
- DTProbLog: the Algorithms
    - Exact Solution Algorithm
    - Approximate Solution Algorithms
- Experiments: Viral Marketing
- Related Work & Conclusions

DTProbLog: the Language


Maggie
Lenny
Homer
Marge
Moe
Bart
Lisa
Apu
Seymour
Ralph


Lenny
Moe
Homer
1
Marge
Maggie
Bart
Lisa
Apu
Seymour
Ralph


Lenny
Moe
0.3
Homer
1
Marge
0.3
Maggie
Apu
Seymour
Bart
0.3
Lisa
Ralph


Lenny
0.14
Moe
0.34
Homer
1
Marge
0.39
Maggie
0.17
Bart
0.41
Lisa
0.25
Apu
0.17
Seymour
0.25
Ralph
0.21

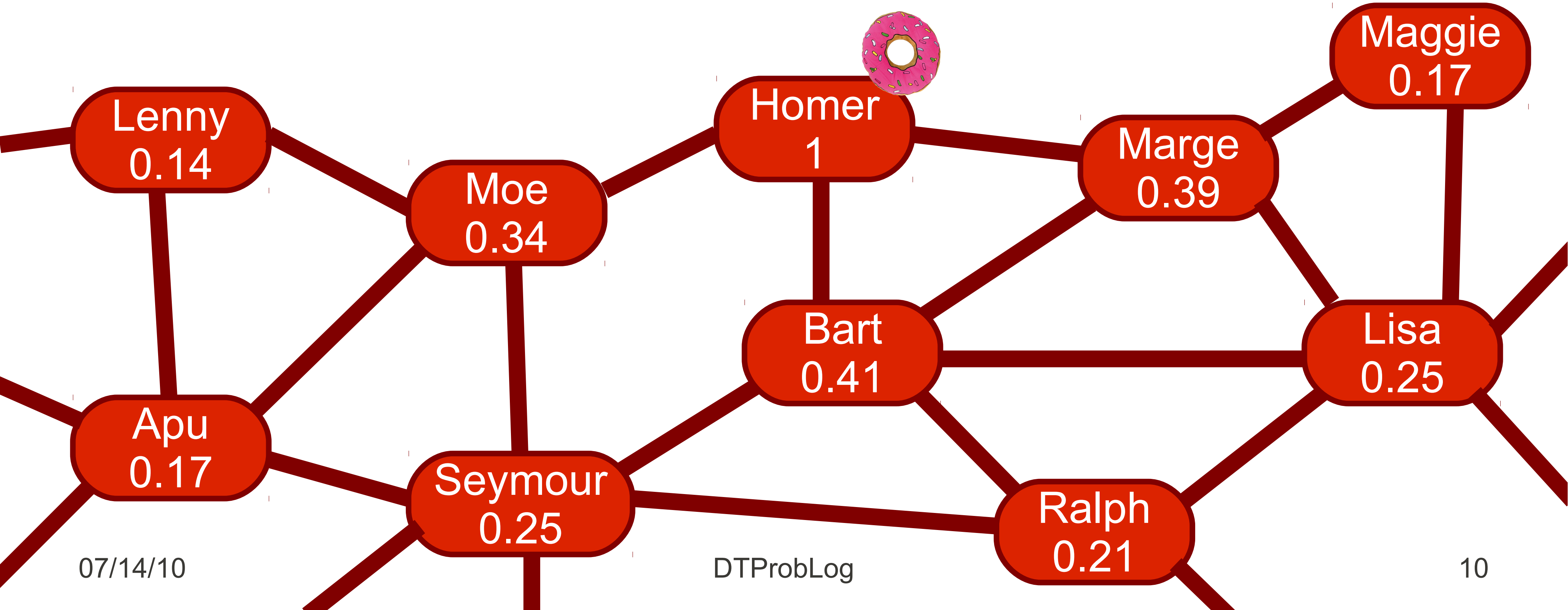
ProbLog

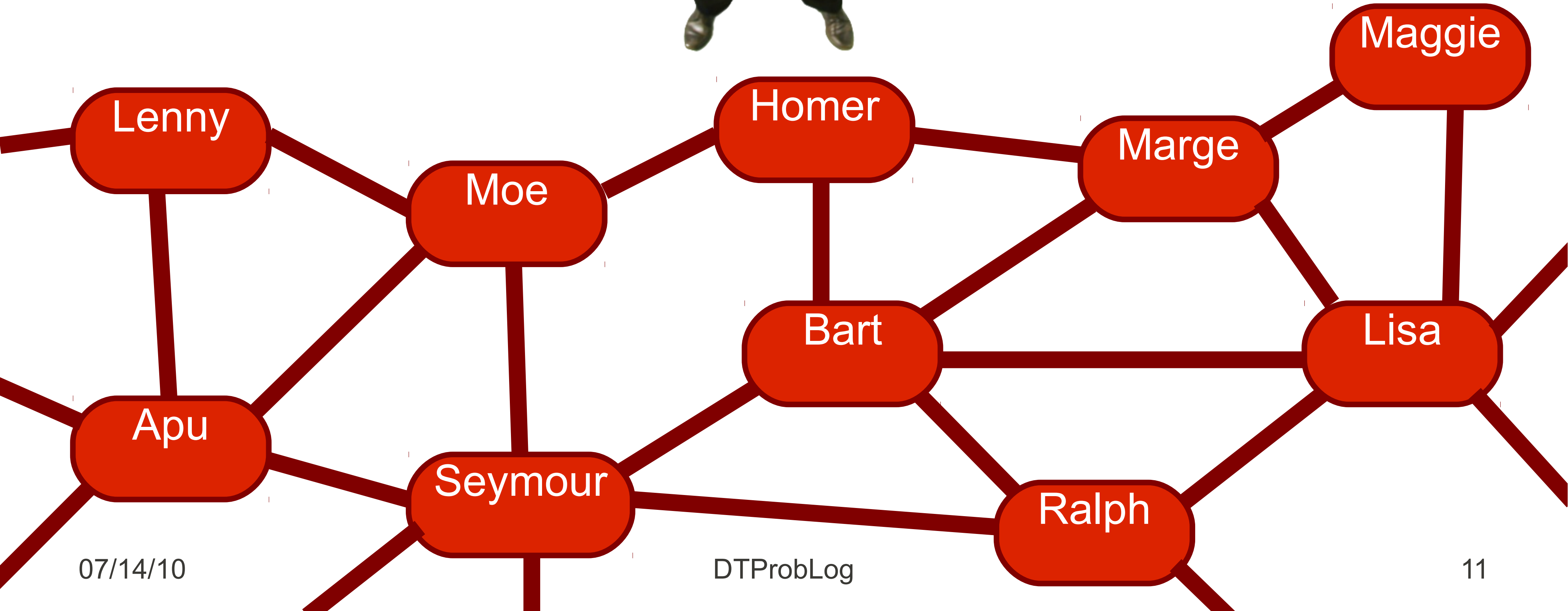
Probabilistic Facts

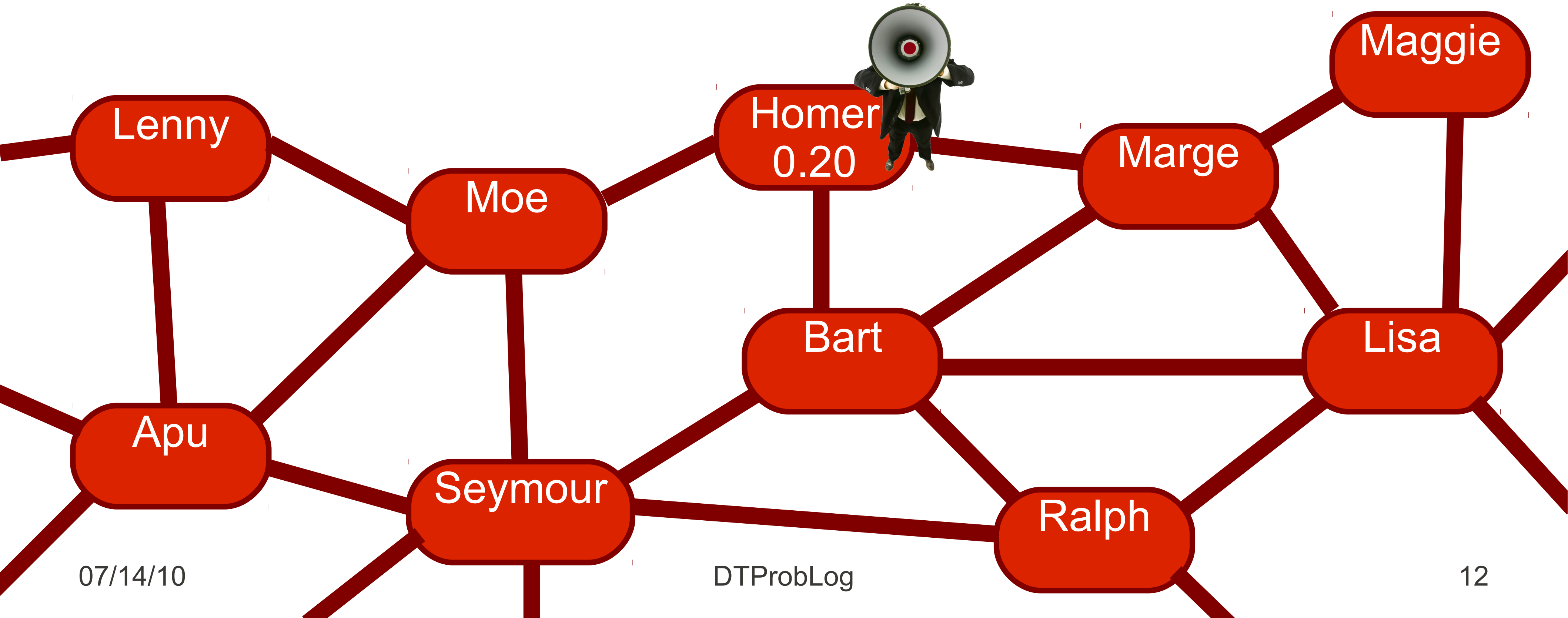
```
0.3 :: buy_trust(_,_).
```

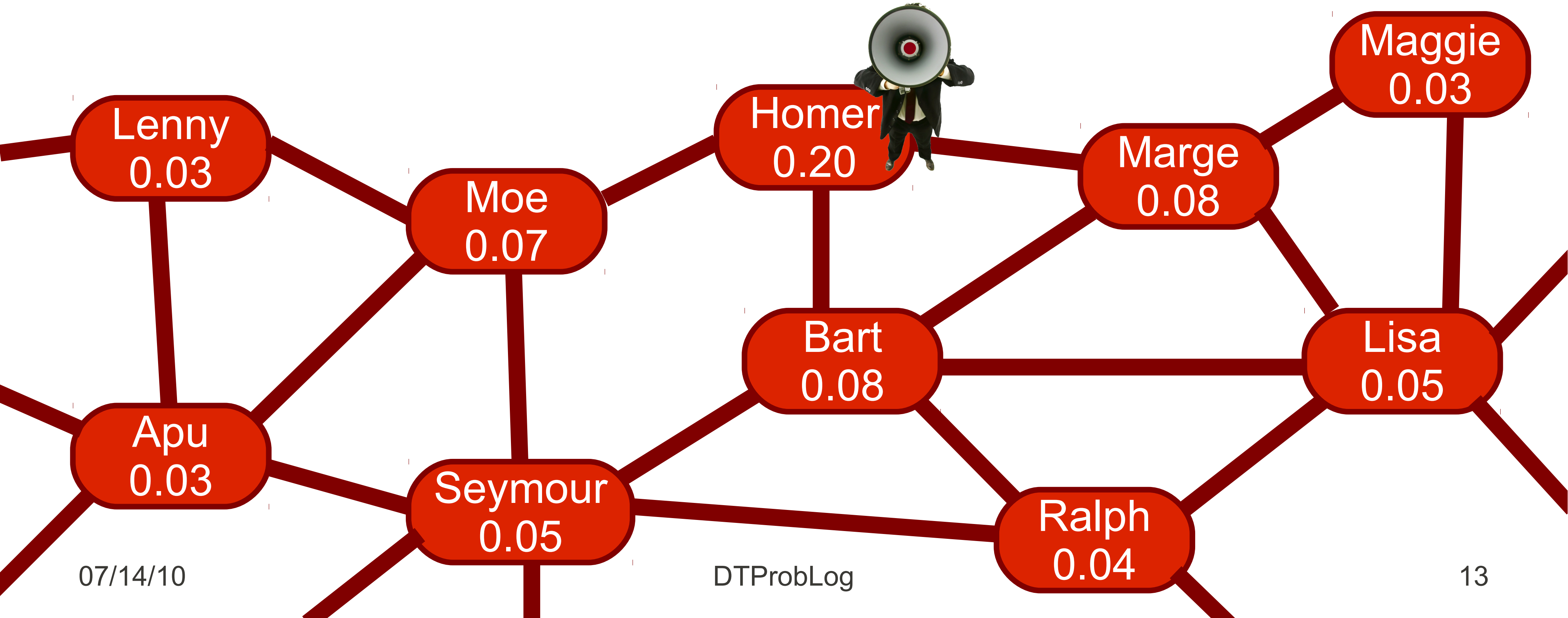
Background Knowledge

```
buys(X) :-
     trusts(X,Y),
     buys(Y),
     buy_trust(X,Y).
```




Maggie
Lenny
Homer
Marge
Moe
Bart
Lisa
Apu
Seymour
Ralph


Lenny
Moe
Homer
0.20
Marge
Maggie
Bart
Lisa
Apu
Seymour
Ralph


Lenny
0.03
Moe
0.07
Homer
0.20
Marge
0.08
Maggie
0.03
Bart
0.08
Lisa
0.05
Apu
0.03
Seymour
0.05
Ralph
0.04

ProbLog

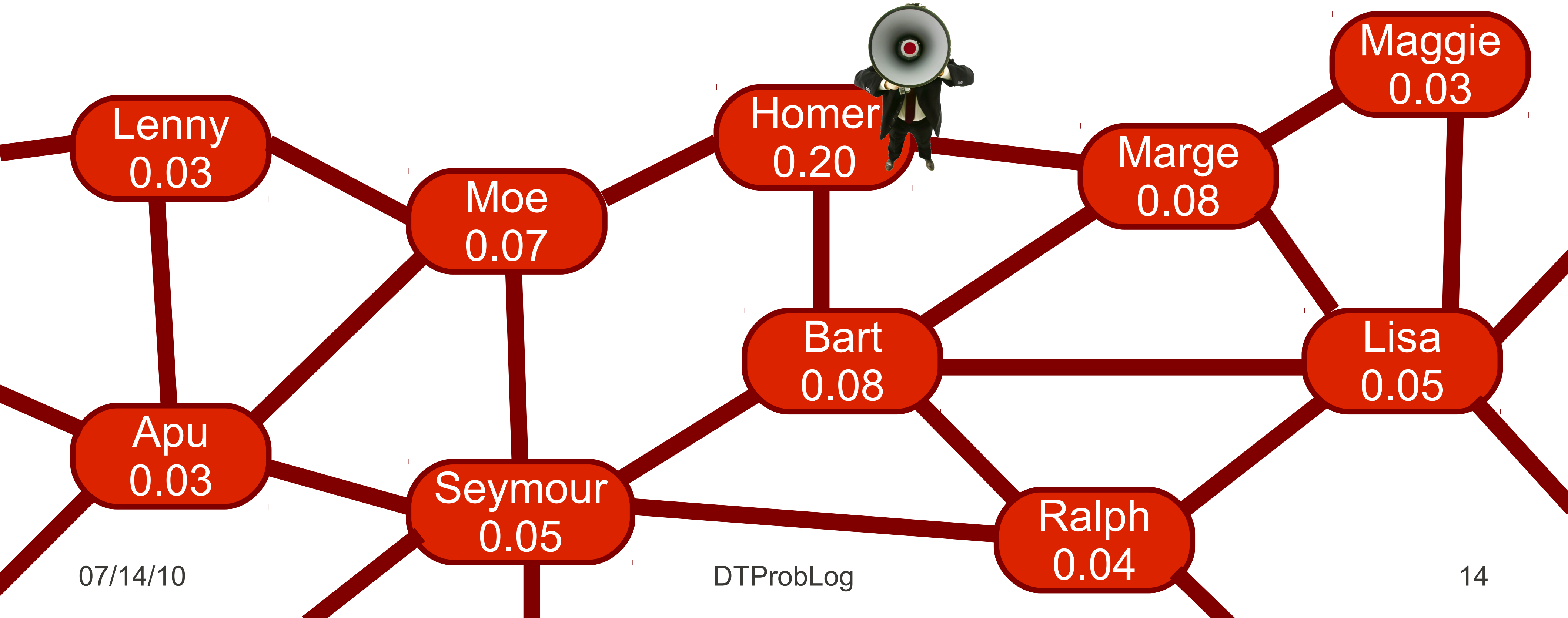
Probabilistic Facts

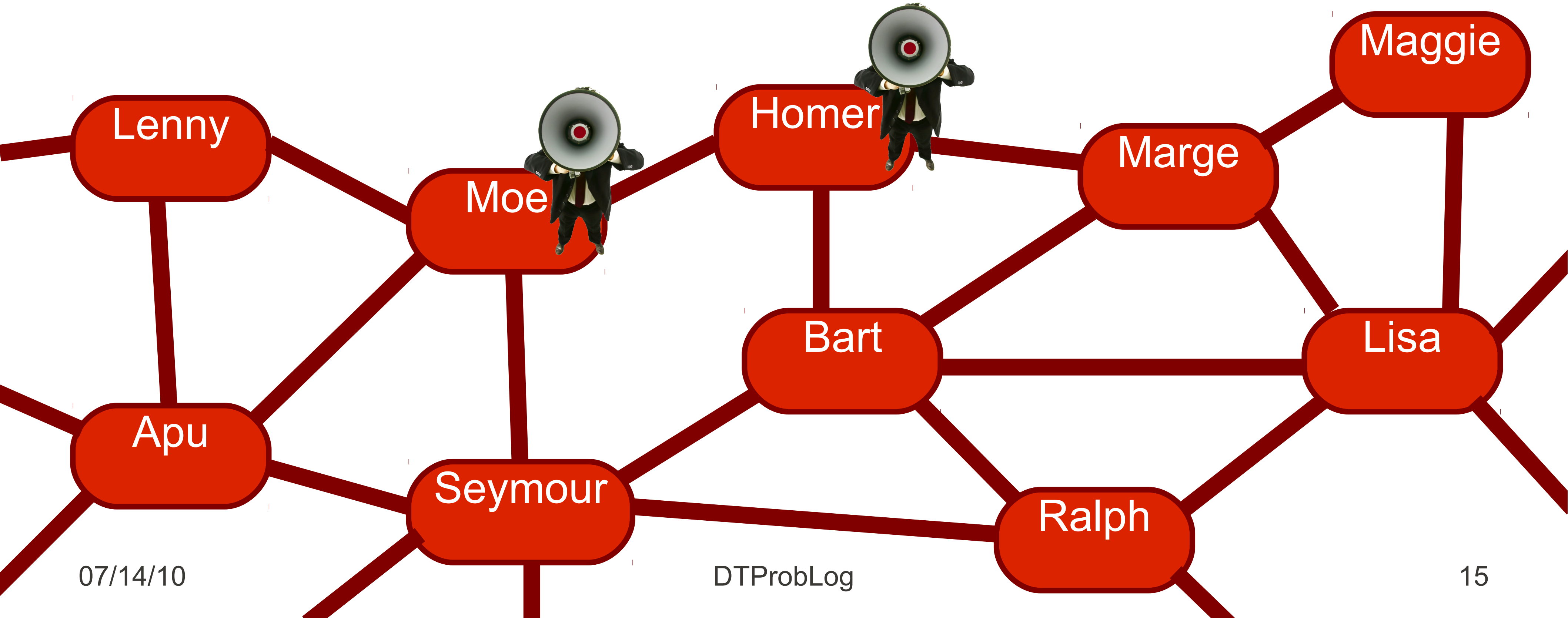
```
0.3 :: buy_trust(_,_).
0.2 :: buy_marketing(_).
```

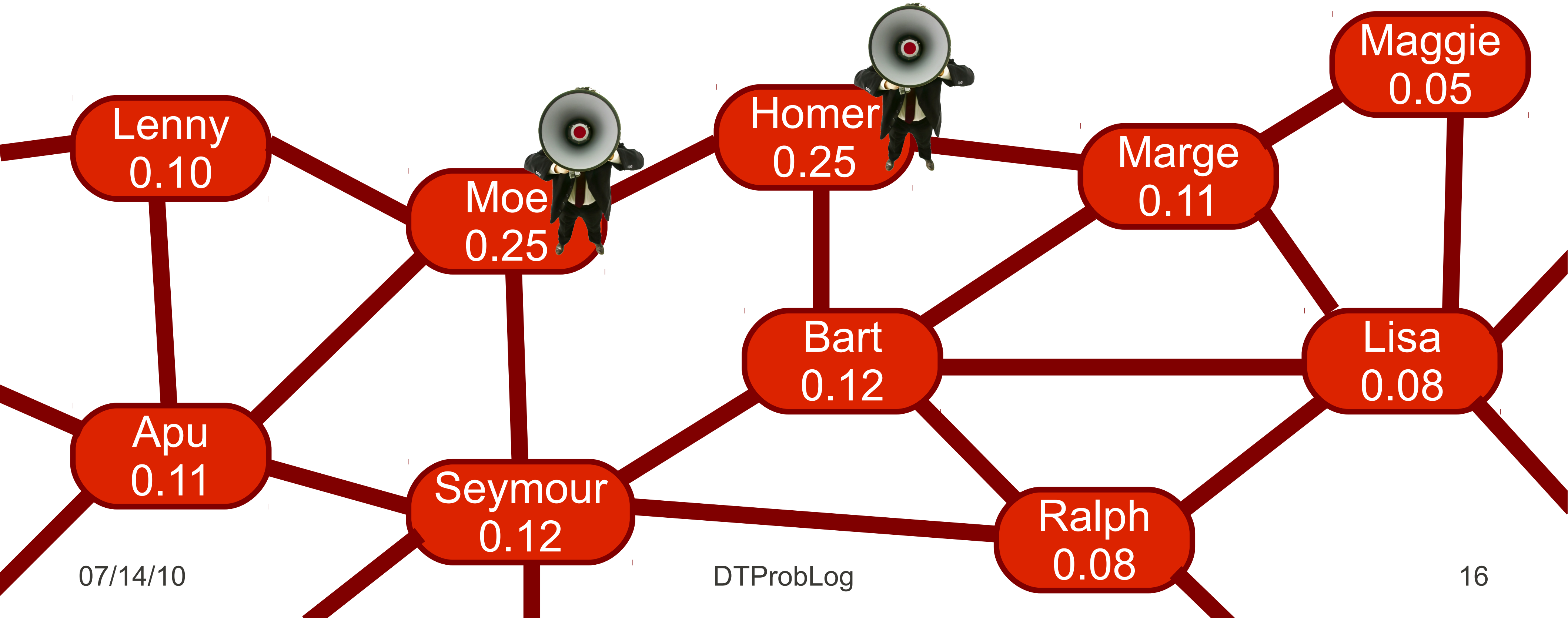
Background Knowledge

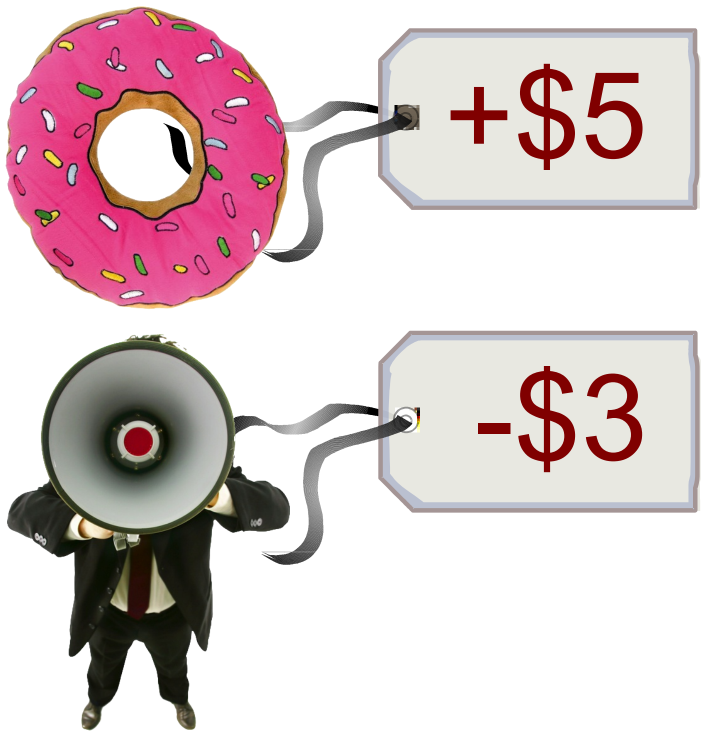
```
buys(X) :-
      trusts(X,Y),
      buys(Y),
      buy_trust(X,Y).
```

```
buys(X) :-
      marketed(X),
      buy_marketing(X).
```

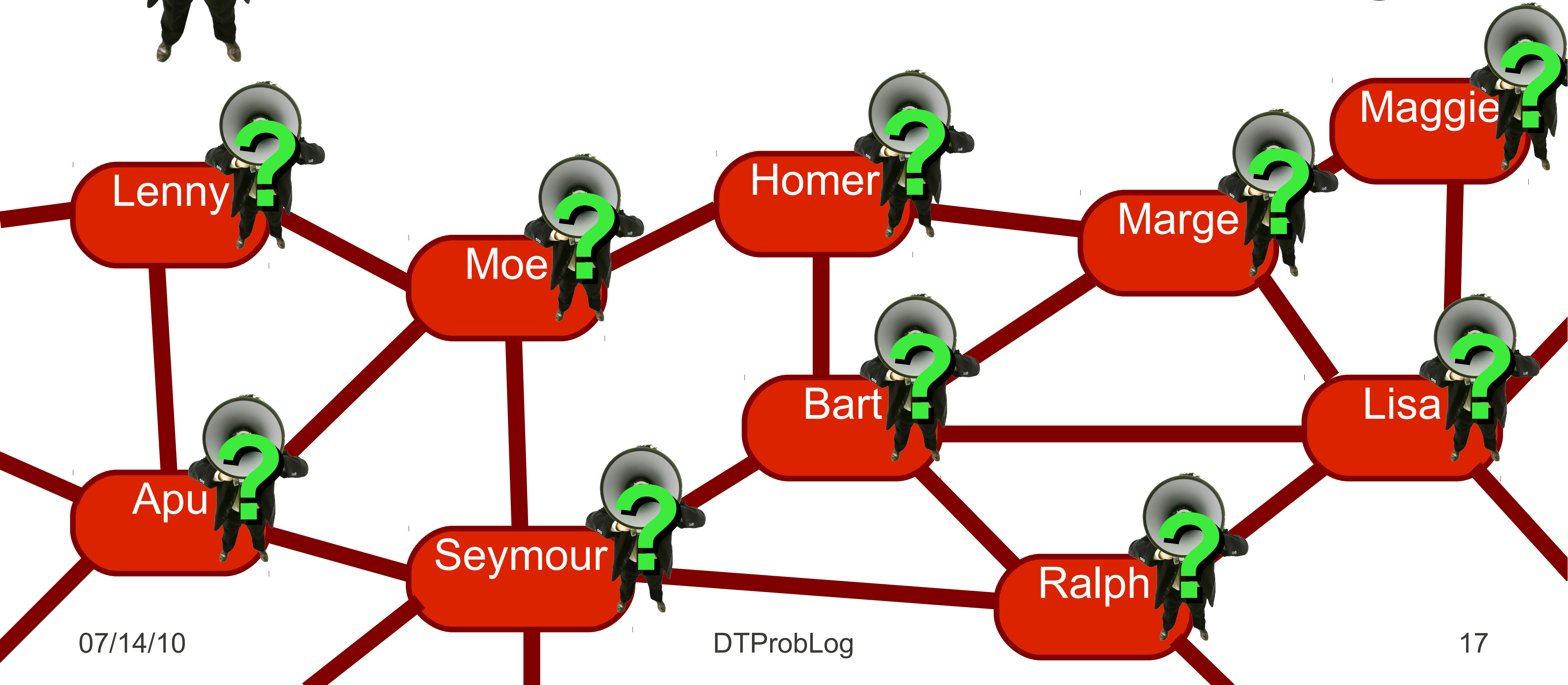


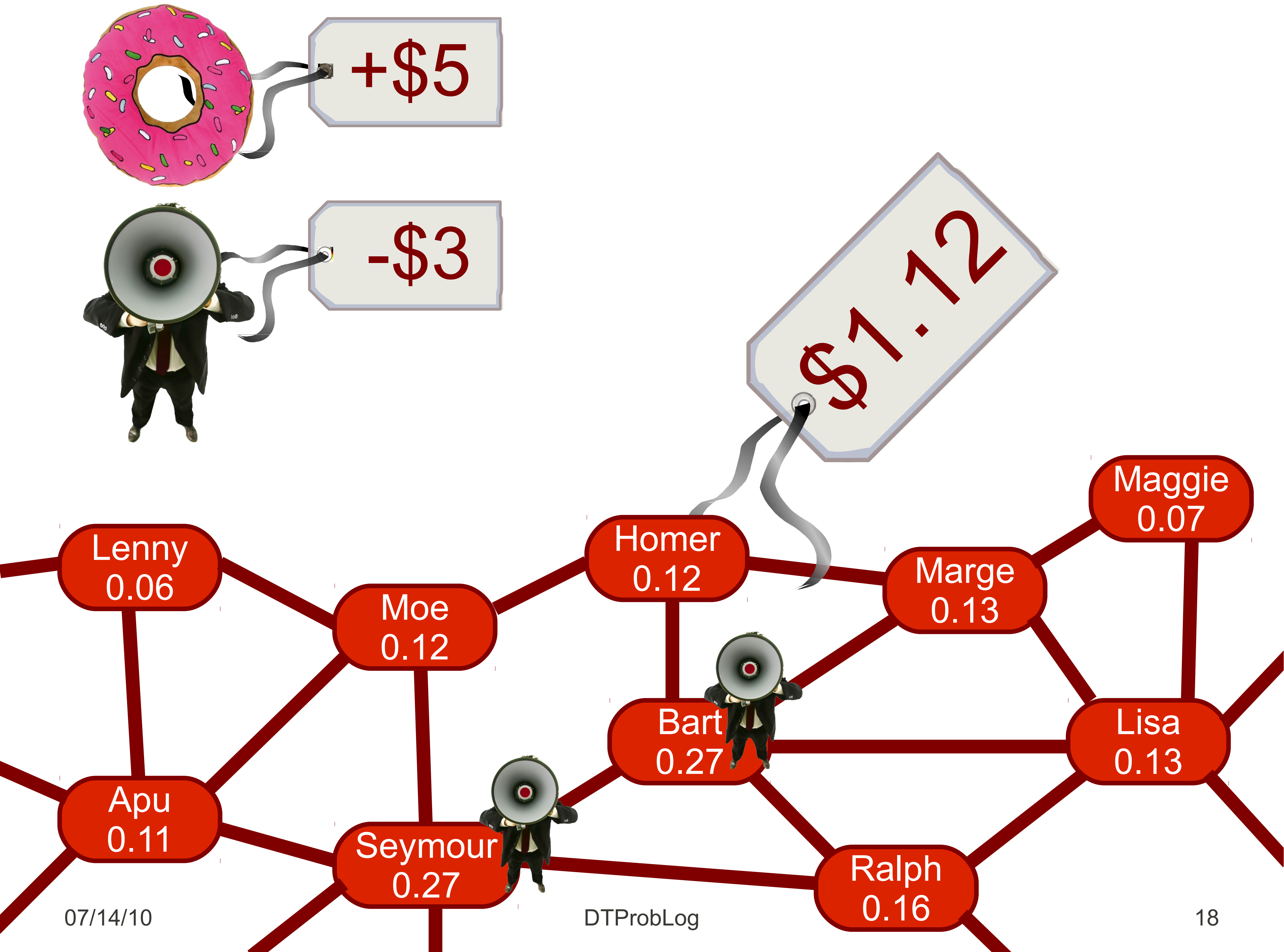

Lenny
Moe
Homer
Marge
Maggie
Bart
Lisa
Apu
Seymour
Ralph


Lenny
0.10
Moe
0.25
Homer
0.25
Marge
0.11
Maggie
0.05
Bart
0.12
Lisa
0.08
Apu
0.11
Seymour
0.12
Ralph
0.08



Which **strategy** gives the **maximum expected utility**?




+$5
-$3
$1.12
Lenny 0.06
Moe 0.12
Homer 0.12
Marge 0.13
Maggie 0.07
Bart 0.27
Lisa 0.13
Apu 0.11
Seymour 0.27
Ralph 0.16

DTProbLog

Probabilistic Facts

...

Background Knowledge

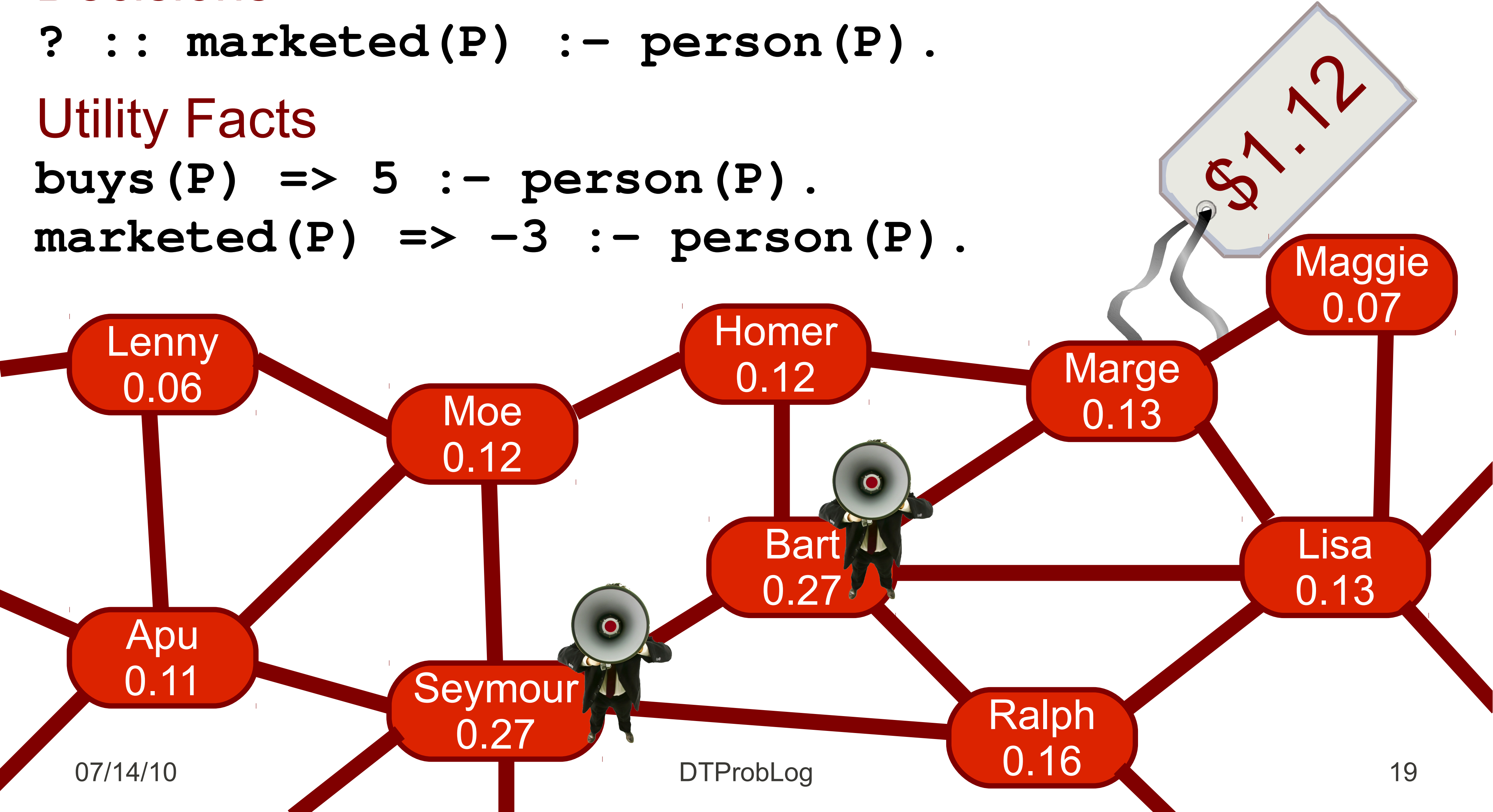
...

Decisions

```
? :: marketed(P) :- person(P).
```

Utility Facts

```
buys(P) => 5 :- person(P).
marketed(P) => -3 :- person(P).
```



DTProbLog: the Algorithms

DTProbLog: the Algorithms

- DTProbLog solves **decision** problems in complex **relational** and **uncertain** environments.
- Exact solution algorithm
  - Extends ProbLog's BDD-based inference
  - Efficient datastructures: BDD and ADD
- Approximate algorithms
  - Local search
  - K-best proofs

Example:
Dressing for unpredictable weather

Decision Facts

```
? :: umbrella.
? :: raincoat.
```

Probabilistic Facts

```
0.3 :: rainy.
0.5 :: windy.
```

Background Knowledge

```
dry :- rainy, umbrella, not(broken_umbrella).
dry :- rainy, raincoat.
dry :- not(rainy).

broken_umbrella :- umbrella, rainy, windy.
```

Utility Facts

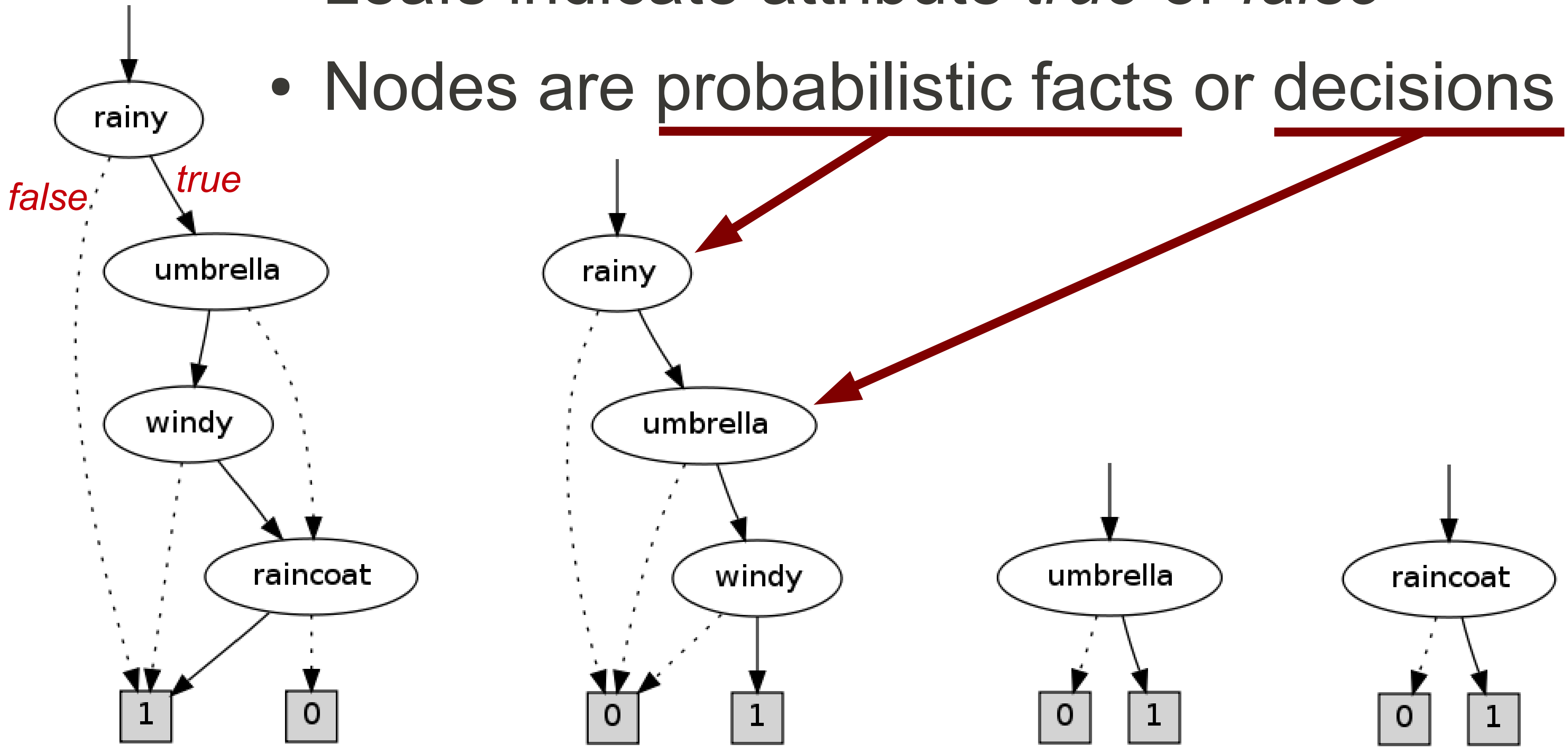
```
umbrella => -2.
raincoat => -20.
```

```
dry => 60.
broken_umbrella => -40.
```

Exact Solution Algorithm

1. Find all **proofs** for each utility attribute (Prolog)
2. Binary decision diagrams from the proofs
3. Algebraic decision diagrams for the **probability** of each attribute
4. Algebraic decision diagrams for the **utility** of each attribute
5. Algebraic decision diagram for the **total utility**

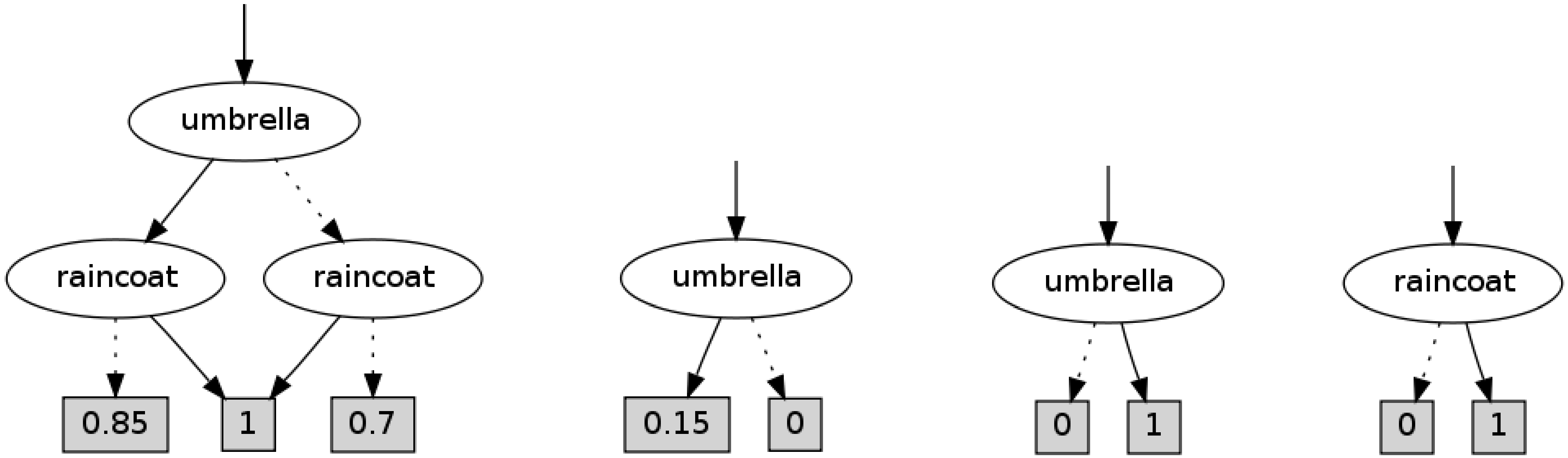
1. Find all **proofs** for each utility attribute (Prolog)
2. Binary decision diagrams from the proofs
   - Leafs indicate attribute *true* or *false*
   - Nodes are probabilistic facts or decisions



```
dry => 60.                     umbrella => -2.
        broken_umbrella => -40.    raincoat => -20.
```

3 Algebraic decision diagrams for the **probability** of each attribute

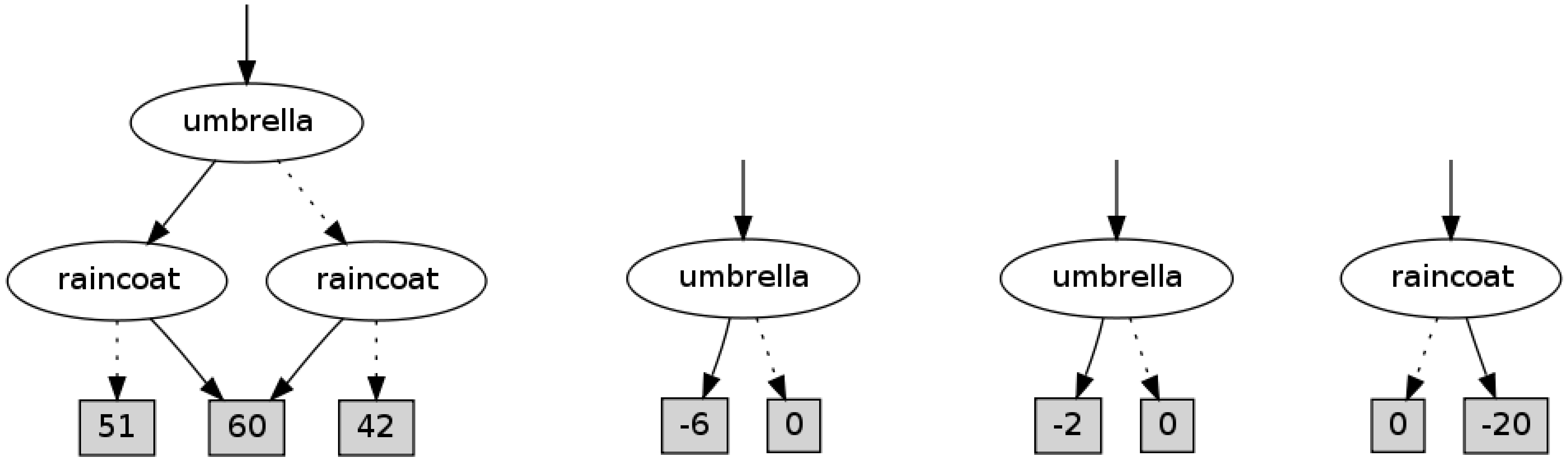
- Probabilistic facts are marginalized out
- Nodes are decisions only
- Leafs are probabilities



```
dry => 60.                        umbrella => -2.
         broken_umbrella => -40.     raincoat => -20.
```

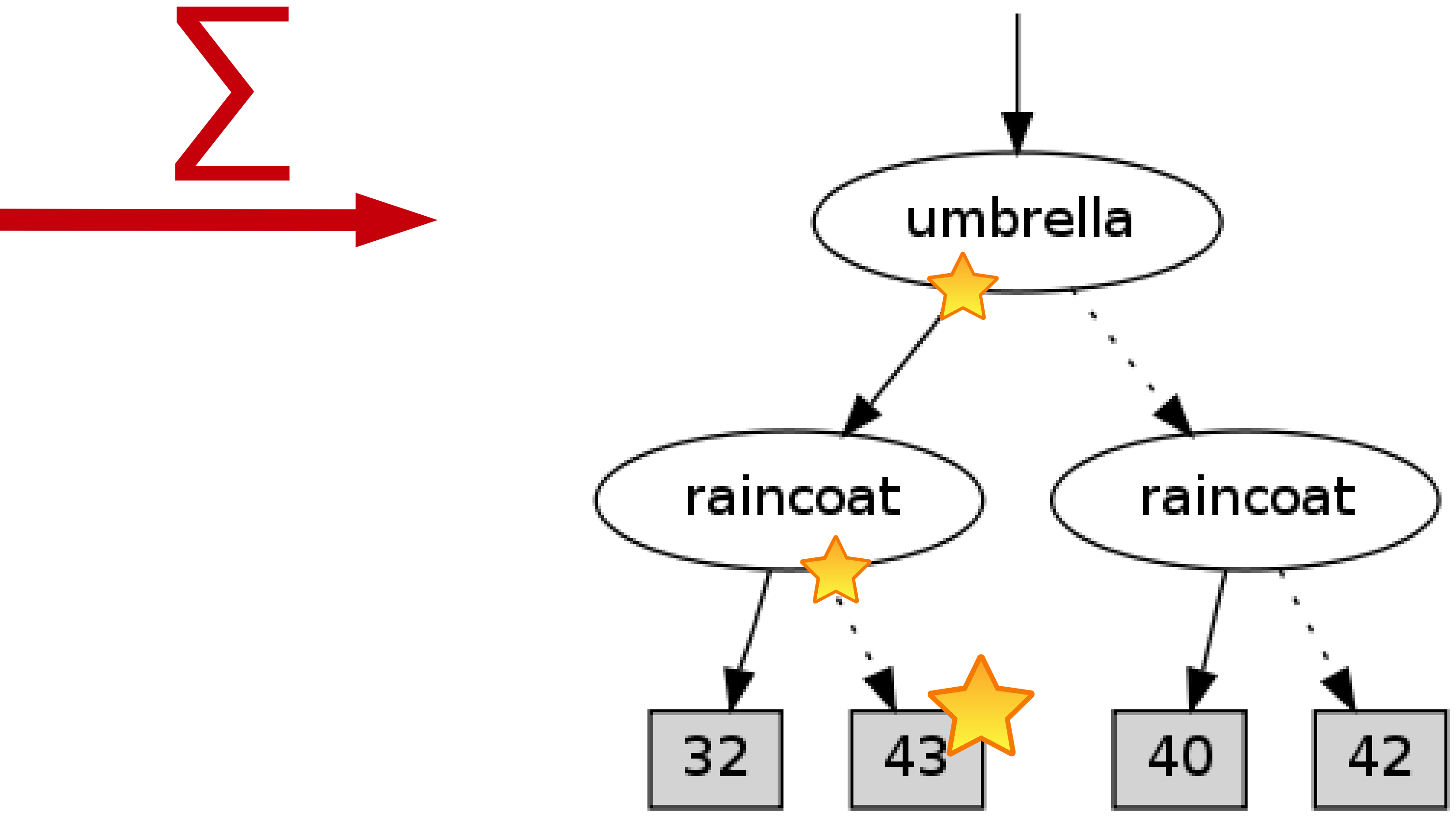
4 Algebraic decision diagrams for the **utility** of each attribute

- Leafs are expected utilities



```
dry => 60.                         umbrella => -2.
        broken_umbrella => -40.     raincoat => -20.
```

5 Algebraic decision diagram for the **total utility**



+ sound pruning (ADD not built entirely in memory)

Approximate Solution: Local Search

1. Find all **proofs** for each utility attribute (Prolog)
2. Binary decision diagrams from the proofs
3. Algebraic decision diagrams for the **probability** of each attribute
4. Algebraic decision diagrams for the **utility** of each attribute
5. Algebraic decision diagram for the **total utility**

Approximate Solution: Local Search

1. Find all **proofs** for each utility attribute (Prolog)
2. Binary decision diagrams from the proofs
3. ~~Algebraic decision diagrams for the **probability** of each attribute~~
4. ~~Algebraic decision diagrams for the **utility** of each attribute~~
5. ~~Algebraic decision diagram for the **total utility**~~

3 Greedy hillclimber search

Approximate Solution: K-best Proofs

1. Find all **proofs** for each utility attribute (Prolog)
2. Binary decision diagrams from the proofs
3. Algebraic decision diagrams for the **probability** of each attribute
4. Algebraic decision diagrams for the **utility** of each attribute
5. Algebraic decision diagram for the **total utility**

Approximate Solution: K-best Proofs

1 ~~Find all **proofs** for each utility attribute (Prolog)~~

1 Find the ***k* most likely** **proofs** for each utility attribute

2 Binary decision diagrams from the proofs

3 Algebraic decision diagrams for the **probability** of each attribute

4 Algebraic decision diagrams for the **utility** of each attribute

5 Algebraic decision diagram for the **total utility**

Experiments: Viral Marketing

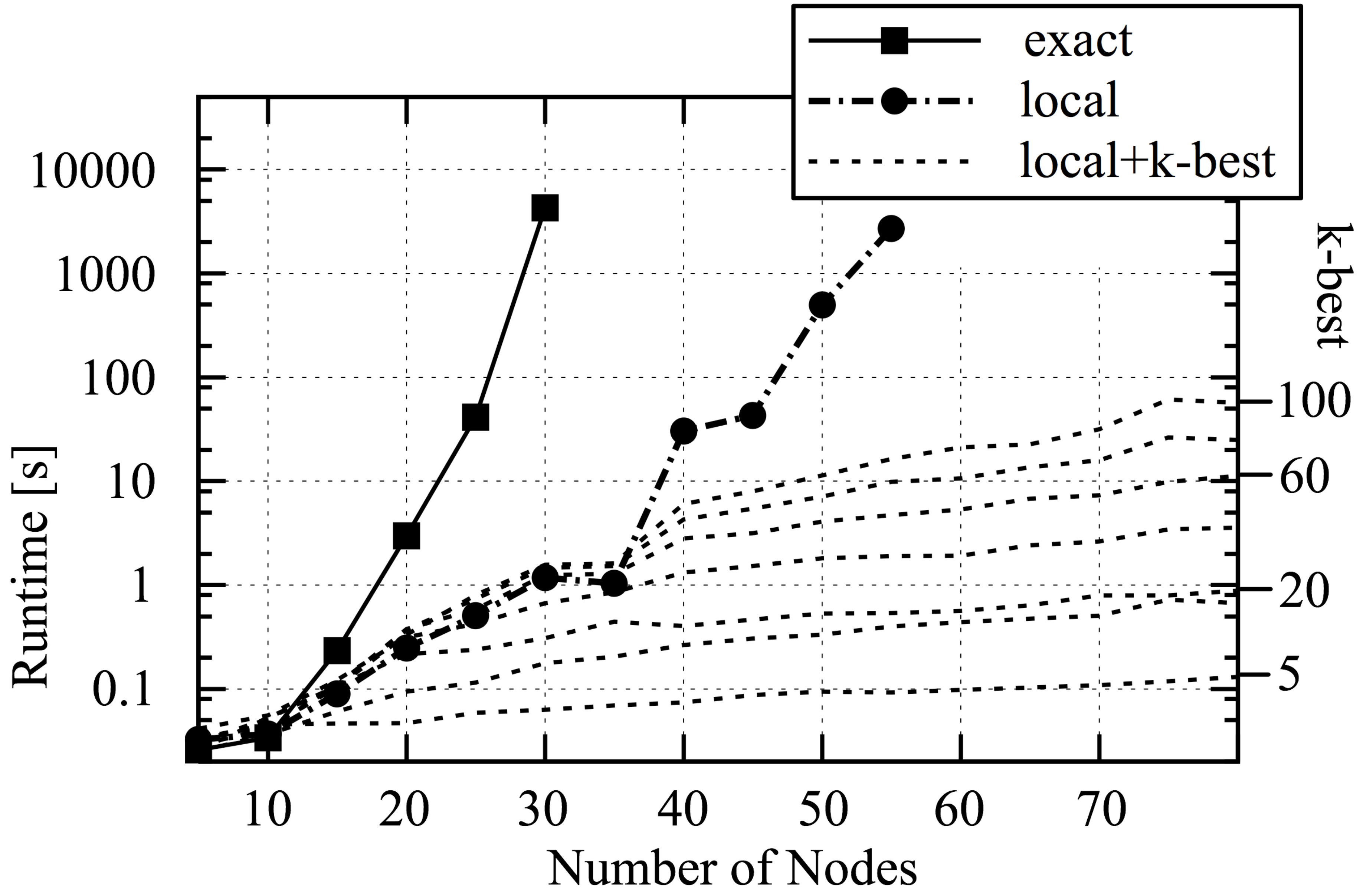
Experiments: Viral Marketing

- Synthetic dataset

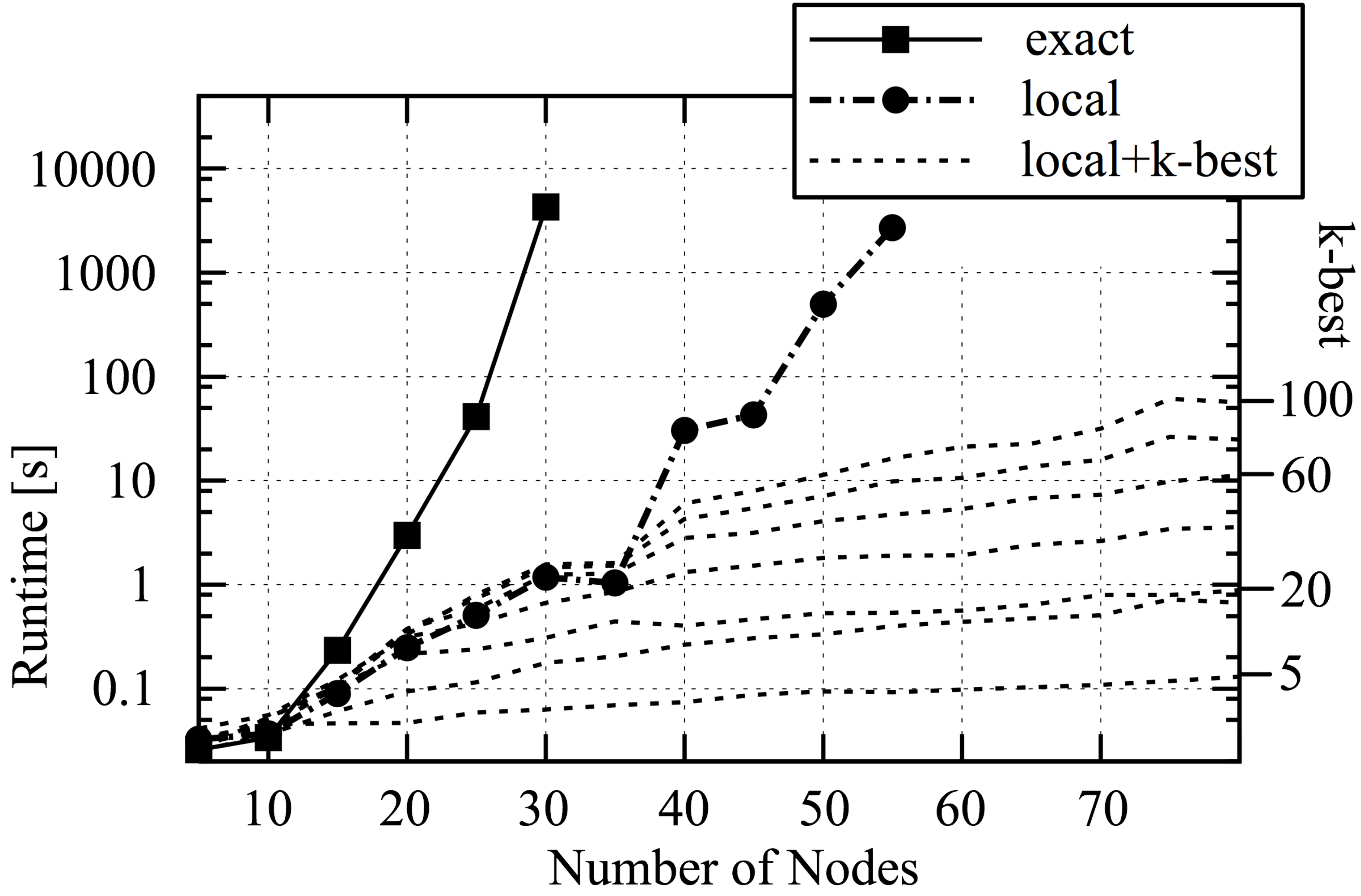
  Random power law graphs of increasing size

- Real-world Epinions dataset [Domingos02]

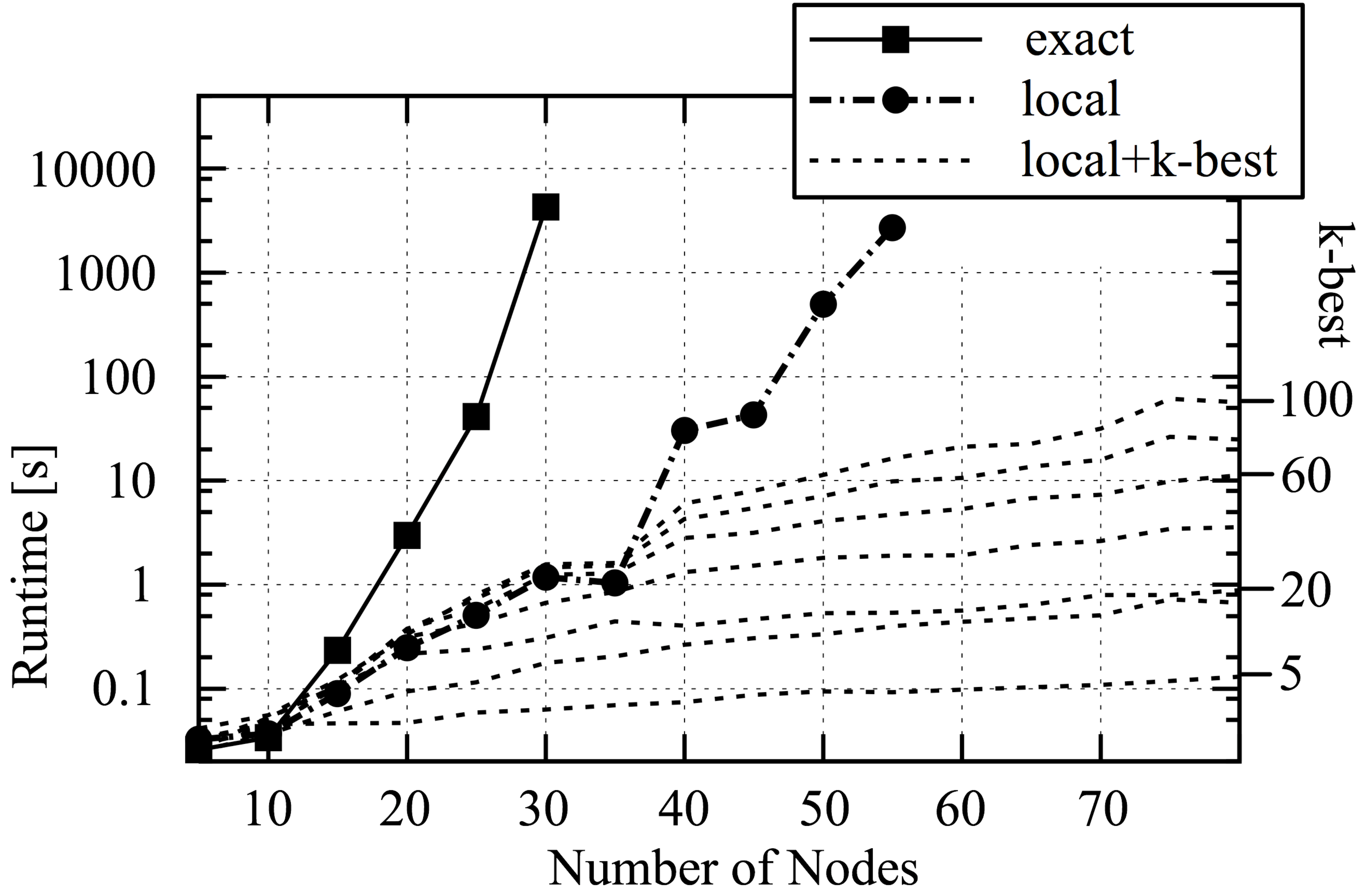
(Q1) Does the **exact** solution algorithm perform better than **naively** calculating the utility of all strategies?



(Q2) What is the difference in runtime and solution quality between **exact** and **local** search?



(Q3) What is the difference in runtime and solution quality between for different values of *k* in **k-best**.



(Q4) Do the algorithms **scale**?

Epinions social network

- 75,000 people
- 500,000 trust relations

YES

- Local search
- 17-best proofs
- Solved in 16 hours

Related Work & Conclusions

Related Work

	Representation		Solution		Evaluation	
	Relational	Probabilities	Global optimum	Local optimum	Exact inference	Approximate inference
Influence Diagrams		✓	✓	?	✓	✓
MLDNs [Nath]	✓	✓		✓		✓
ICL [Poole]	✓	✓	?	?	?	?
DTLPs [Chen]	✓	✓				
DTProbLog	✓	✓	✓	✓	✓	✓

Summary

- DTProbLog, the programming language
  - Probabilistic Prolog
  - Decisions
  - Utilities: rewards or costs attached to goals
- Solution algorithms
  - Exactly
  - Approximately
- Experiments
  - Effective
  - Scale well

Ongoing and Future Work

- Sequential decision problems
  - Easy to represented in DTProbLog
  - Bad fit for solution algorithms
- Solvers
  - Integer linear programming
  - Bounded approximation
  - Monte-Carlo
- Lifting (many BDDs have same structure)
- Learning DTProbLog programs
- Inverse reinforcement learning

Thank You!

Viral Marketing

Decisions

```
? :: marketed(P) :- person(P).
```

Probabilistic Facts

```
0.3 :: buy_trust(_,_).
0.2 :: buy_marketing(_).
```

Background Knowledge

```
buys(X) :-                        buys(X) :-
    trusts(X,Y),                      marketed(X),
    buys(Y),                          buy_marketing(X).
    buy_trust(X,Y).
```

Utility Facts

```
buys(P) => 5 :- person(P).
marketed(P) => -3 :- person(P).
```