The max-min-hill-climbing algorithm

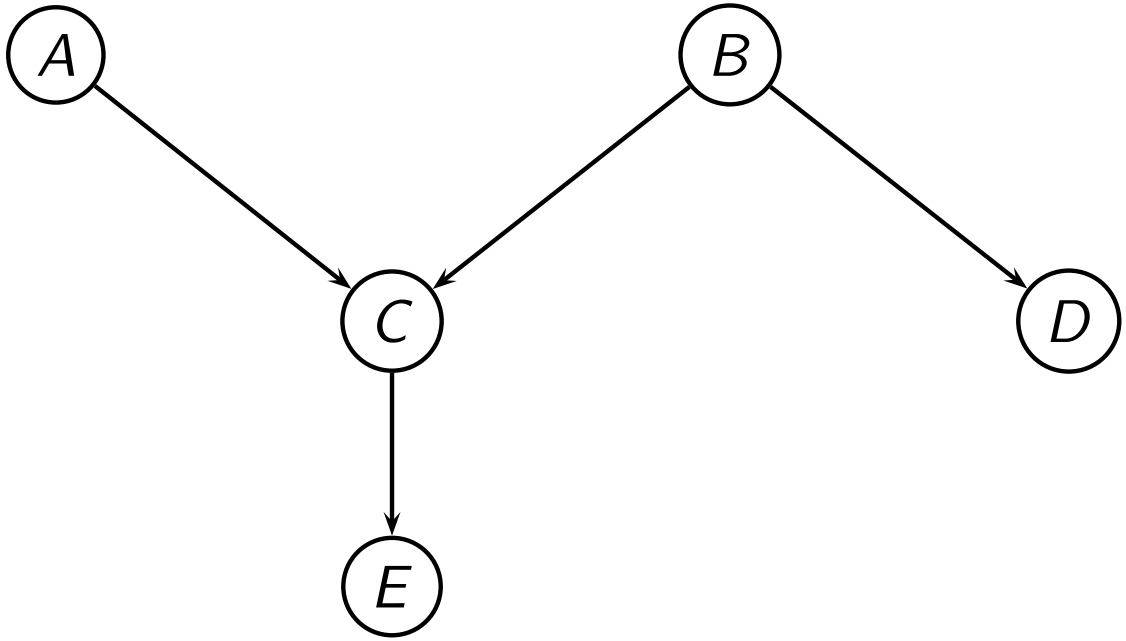
Michael Bauer

M.Sc. Comp. Science

July 15, 2014

Definition

A Bayesian Network is a directed acyclic graph (DAG) whose nodes are random variables and edges represent conditional dependencies. If two random variables are connected they are said to be dependent. If there is no connection they are said to be conditional independent.
For instance, we say: "A and B are conditional independent given C".



- directed edges
- free of cycles
- random variable is represented as a node
- edges encode dependencies

Predicting the effect of missense mutations on protein function: analysis with Bayesian networks

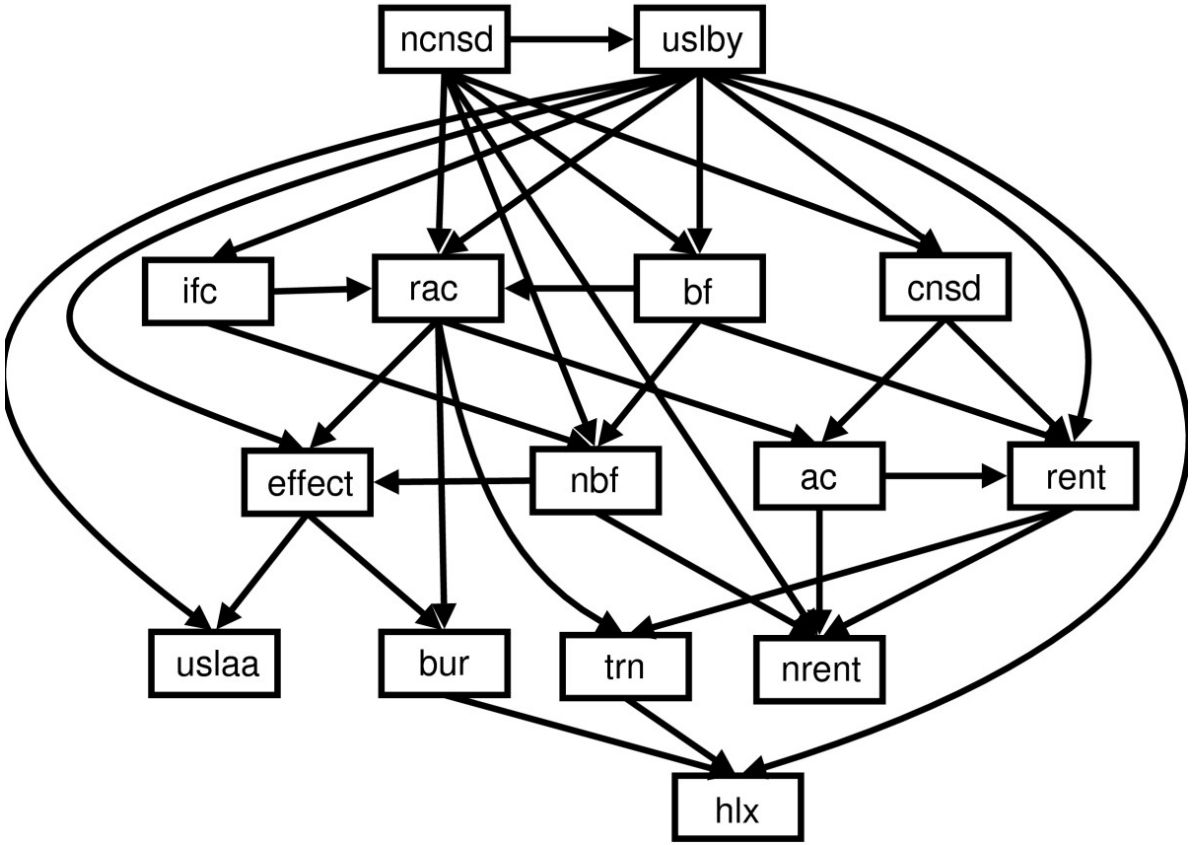


Figure: http://www.biomedcentral.com/1471-2105/7/405/figure/F2?highres=y (by Chris J Needham1, James R Bradford, Andrew J Bulpitt, Matthew A Care and David R Westhead)

Bayesian Networks in sports and medicine

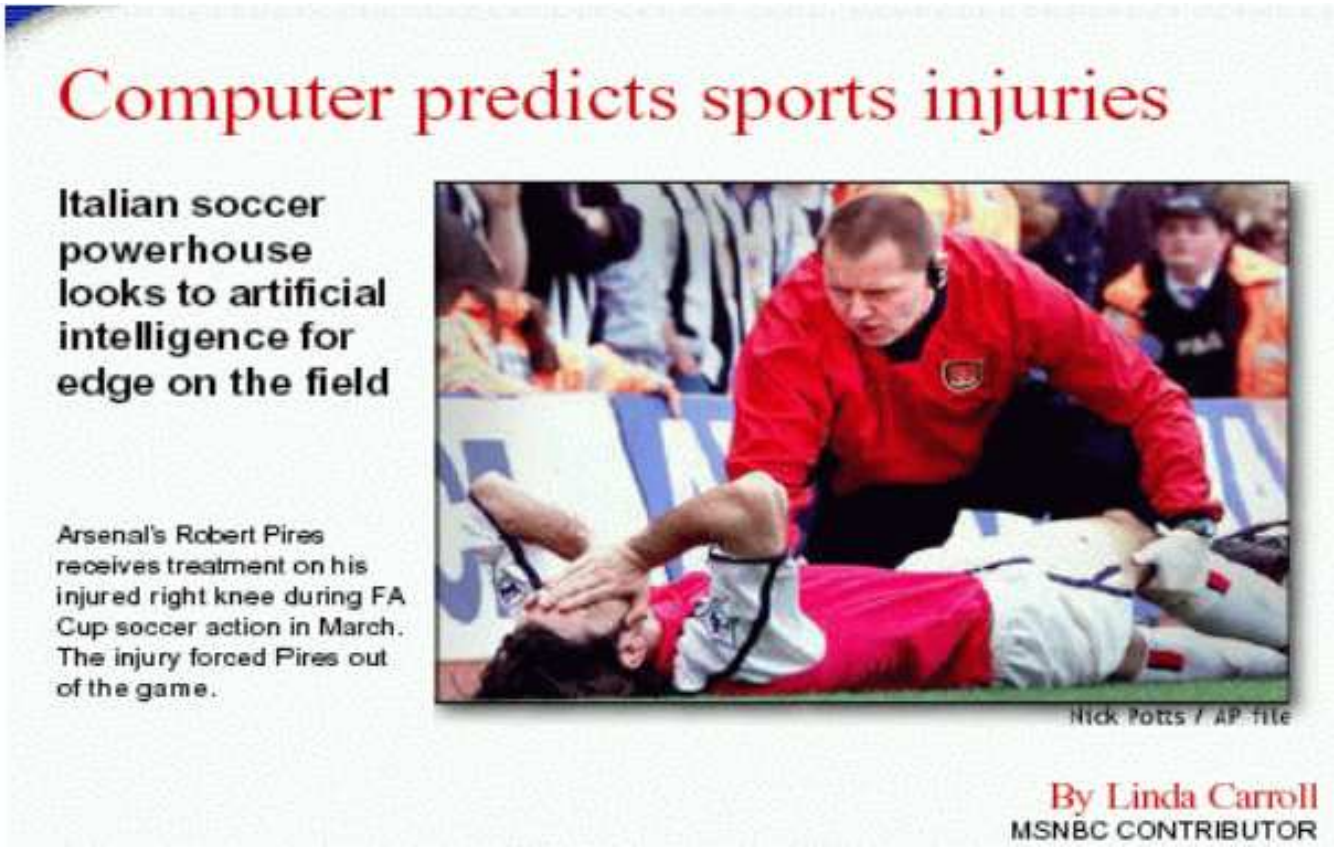
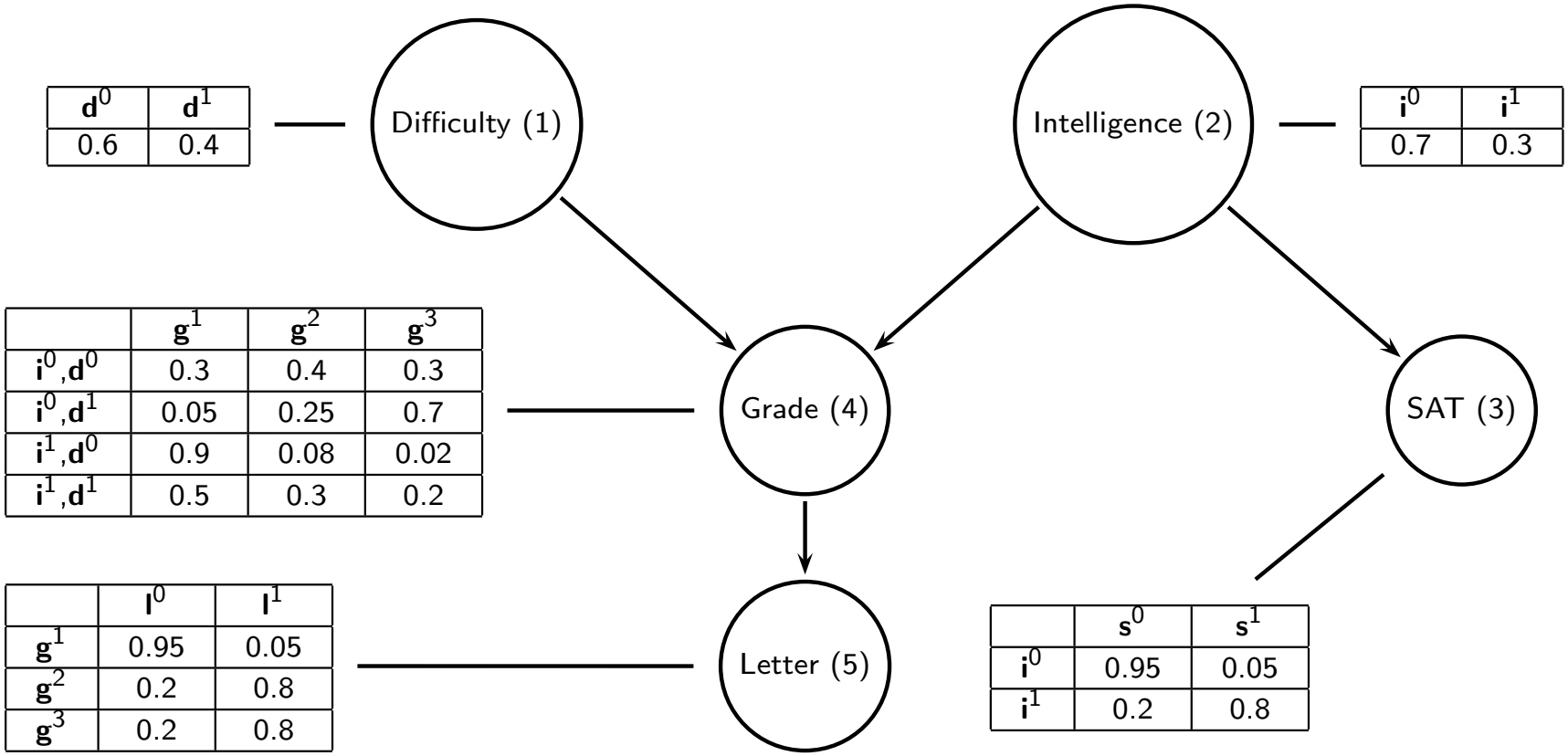


Figure: http://www-ekp.physik.uni-karlsruhe.de/županc/WS1011/docs/Datenanalyse2010_3.pdf



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

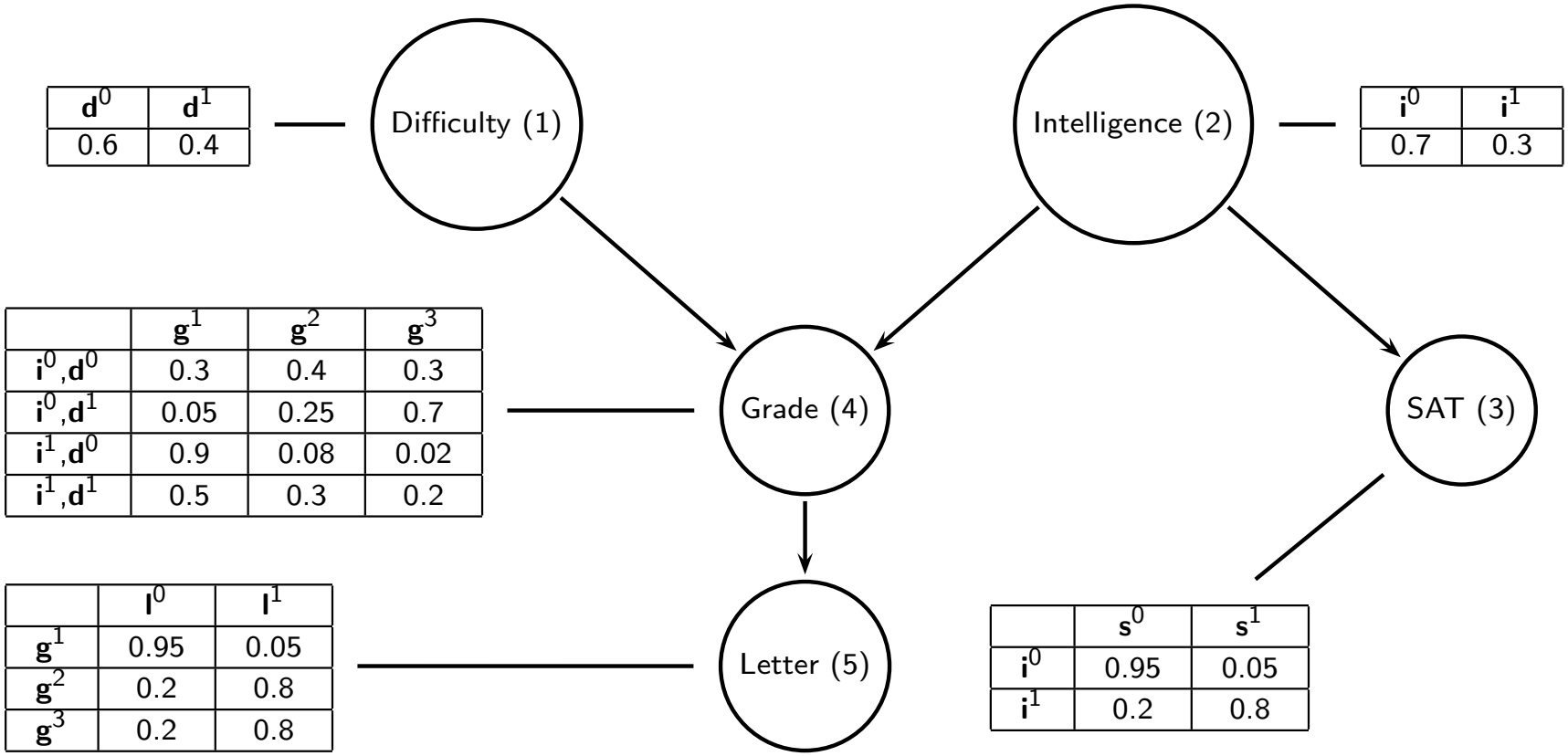
	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

	l^0	l^1
g^1	0.95	0.05
g^2	0.2	0.8
g^3	0.2	0.8

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

```
> head(dataFrame)
     diff int SAT gra let
[1,]    1   2   2   1   2
[2,]    1   1   1   1   2
[3,]    1   1   1   3   1
[4,]    2   2   2   2   2
[5,]    1   1   1   1   2
[6,]    2   1   1   3   1
```

Figure: The data we observe from following the rules above.



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

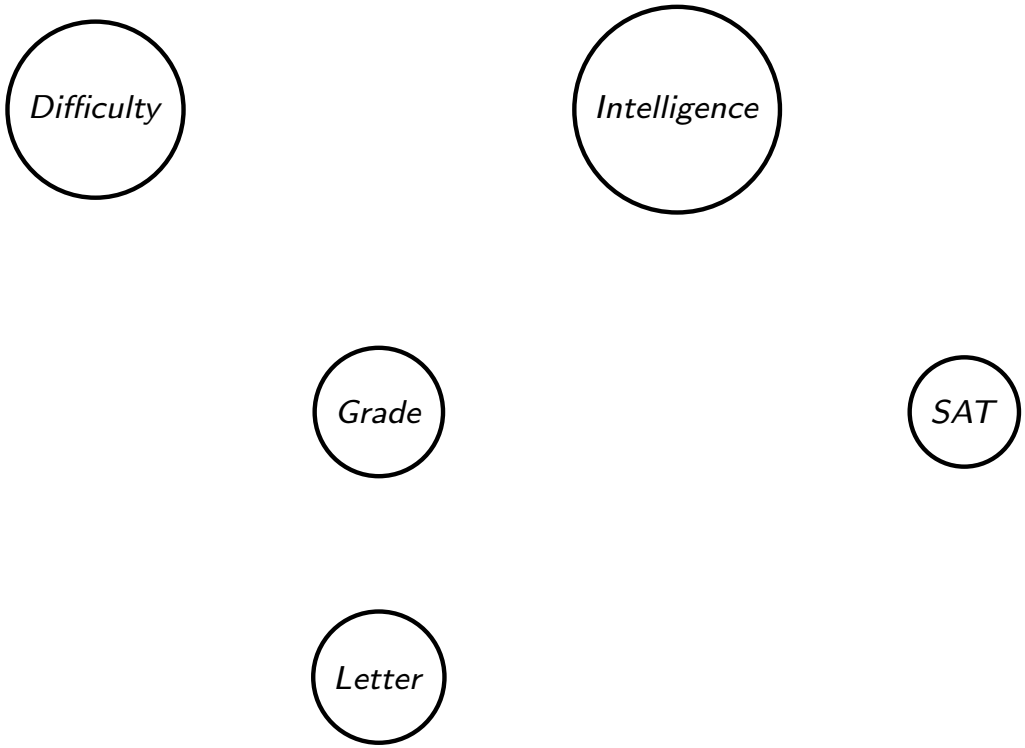
	l^0	l^1
g^1	0.95	0.05
g^2	0.2	0.8
g^3	0.2	0.8

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

```
> head(dataFrame)
     diff int SAT gra let
[1,]    1   2   2   1   2
[2,]    1   1   1   1   2
[3,]    1   1   1   3   1
[4,]    2   2   2   2   2
[5,]    1   1   1   1   2
[6,]    2   1   1   3   1
```

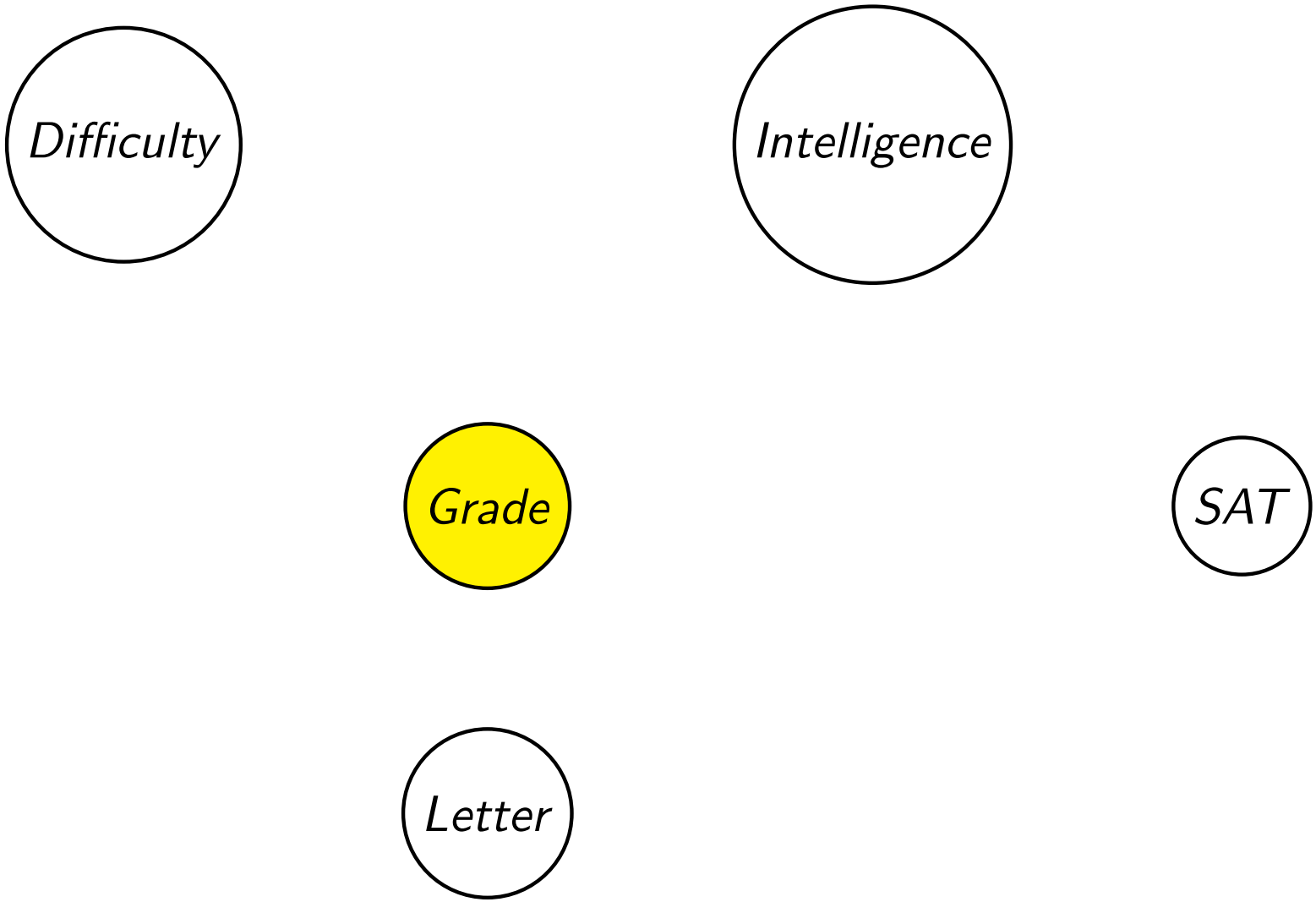
Figure: The data we observe from following the rules above.

Empty graph without any edges

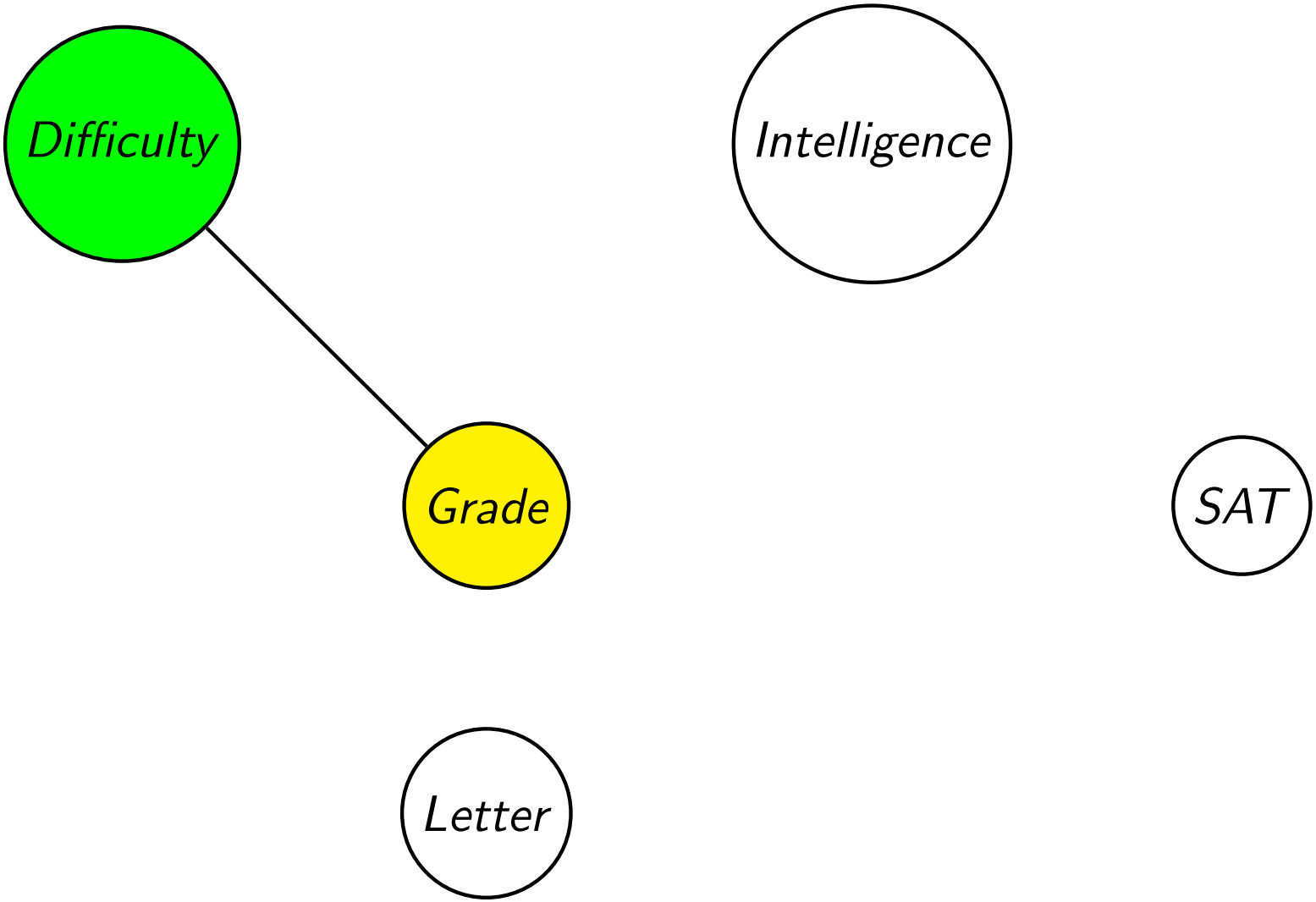



```
> head(dataFrame)
     diff int SAT gra let
[1,]    1   2   2   1   2
[2,]    1   1   1   1   2
[3,]    1   1   1   3   1
[4,]    2   2   2   2   2
[5,]    1   1   1   1   2
[6,]    2   1   1   3   1
```

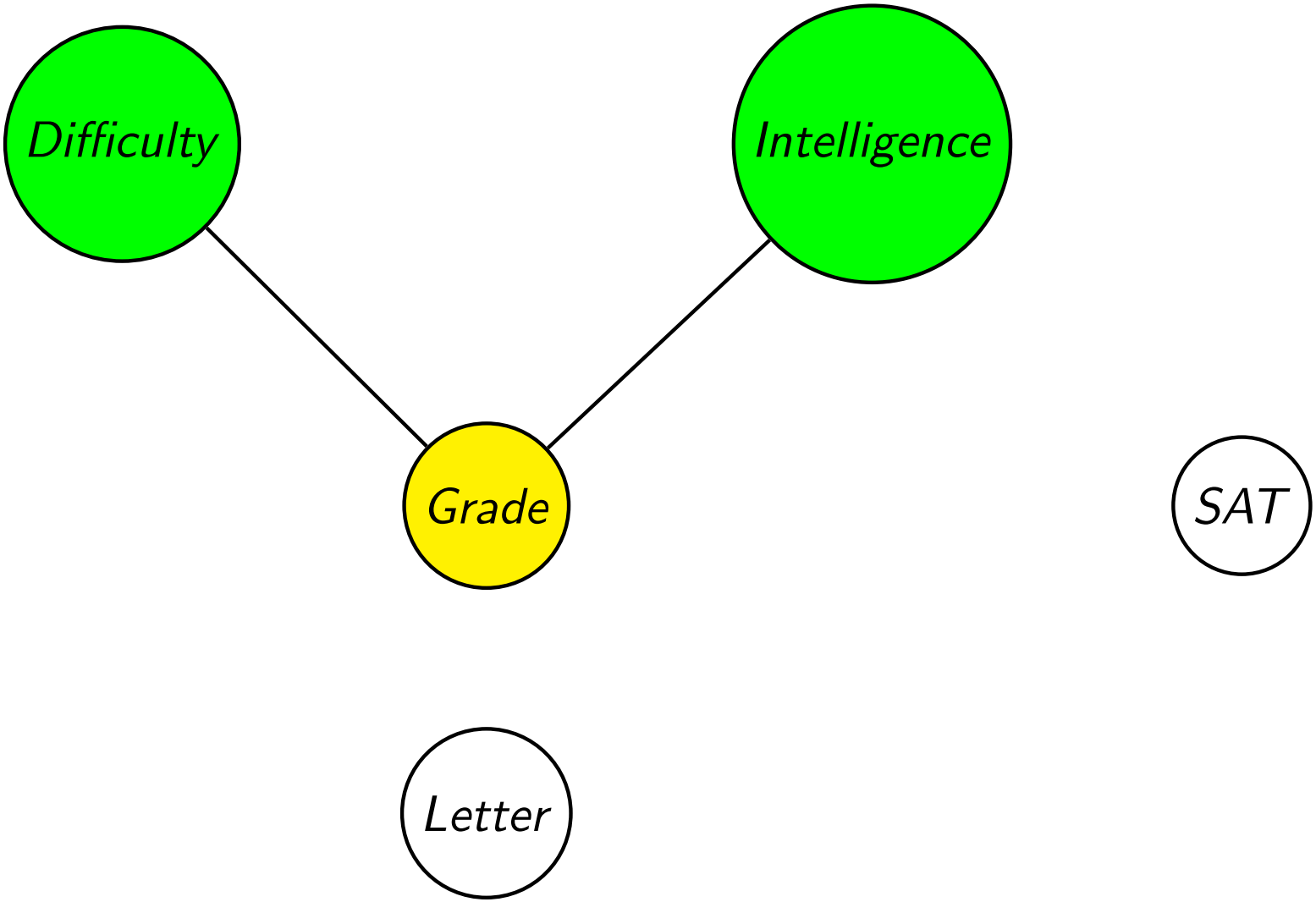
One iteration for the "Grade" node



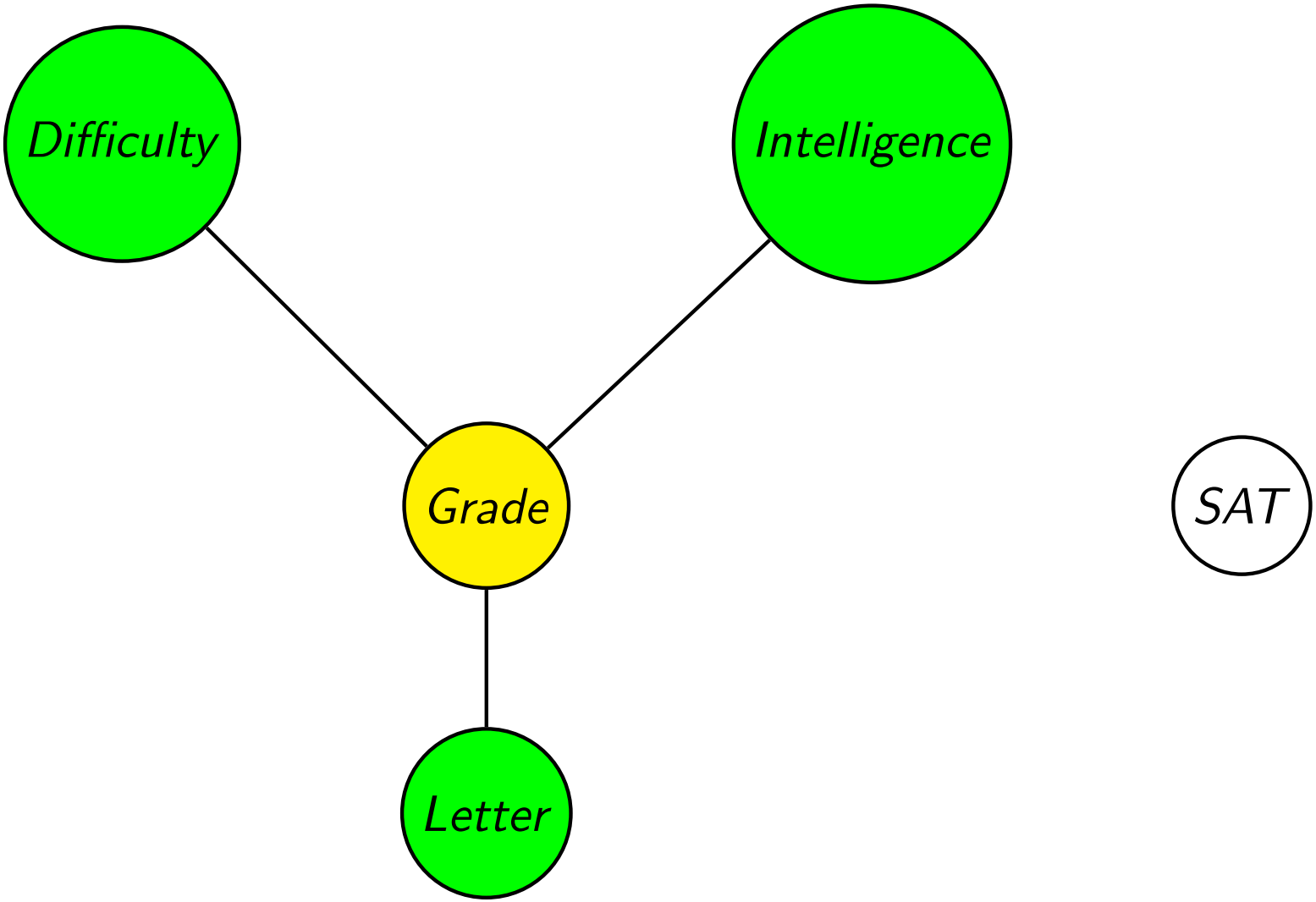
One iteration for the "Grade" node



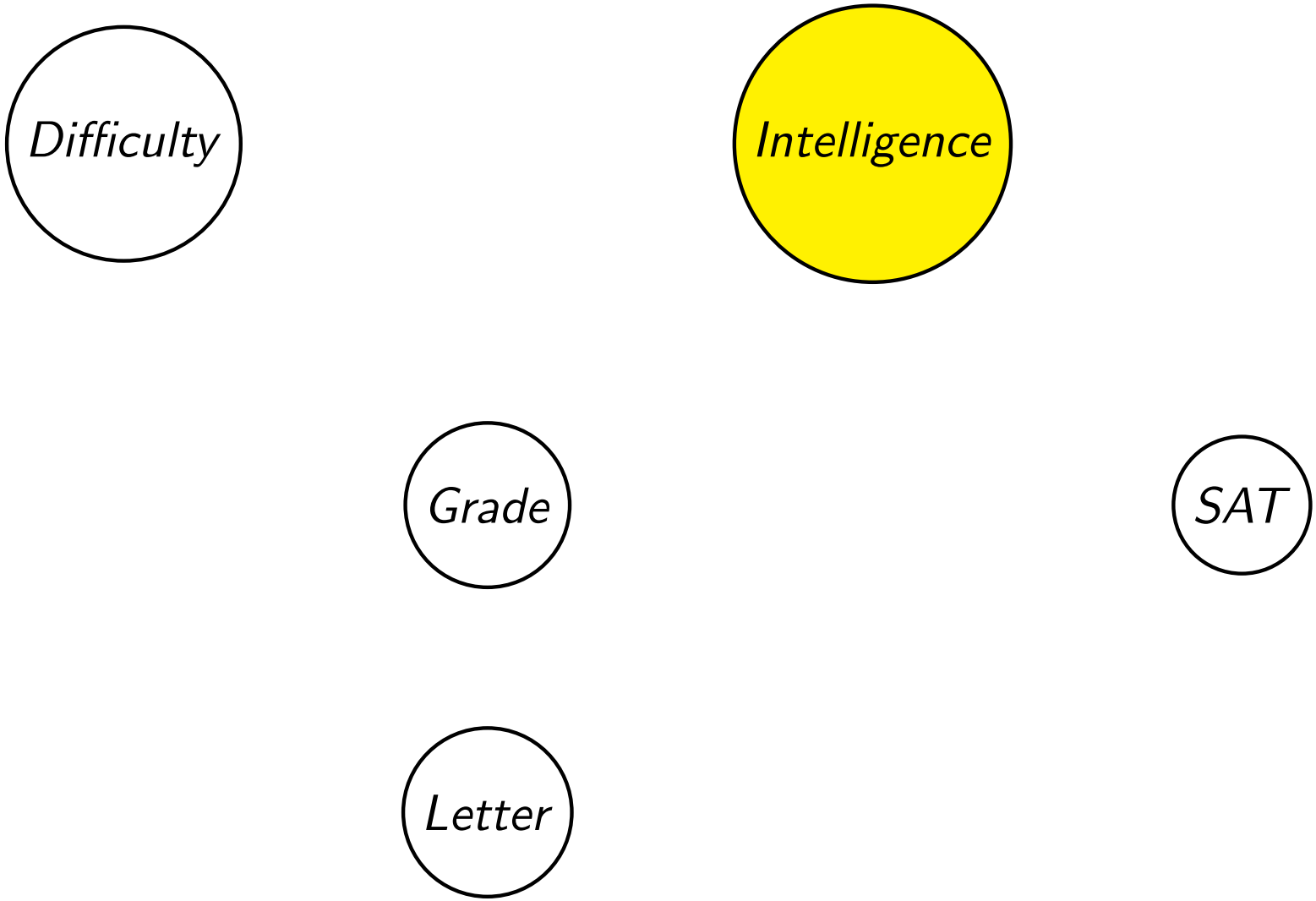
One iteration for the "Grade" node

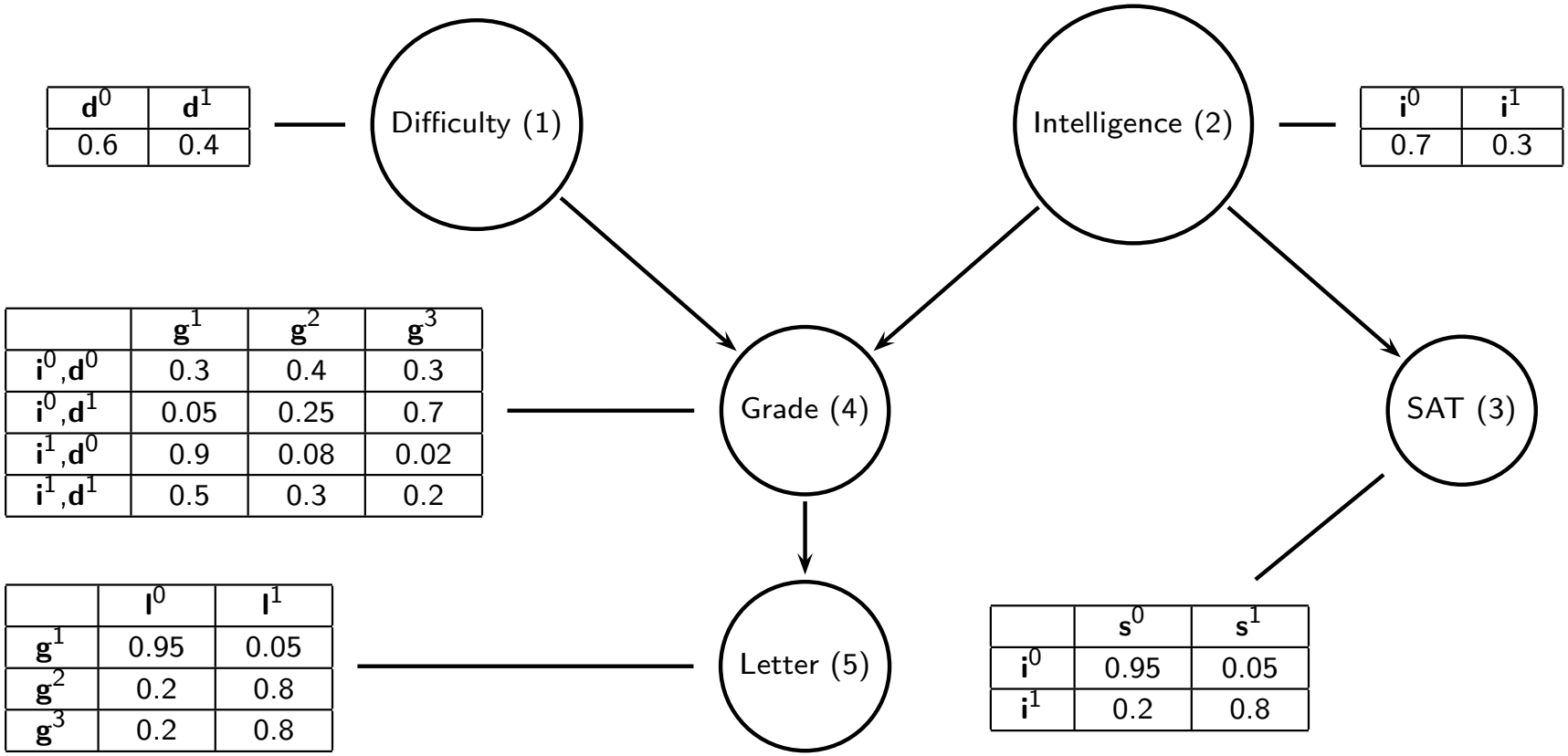


All parents or children are found



Start new iteration





d^0	d^1
0.6	0.4

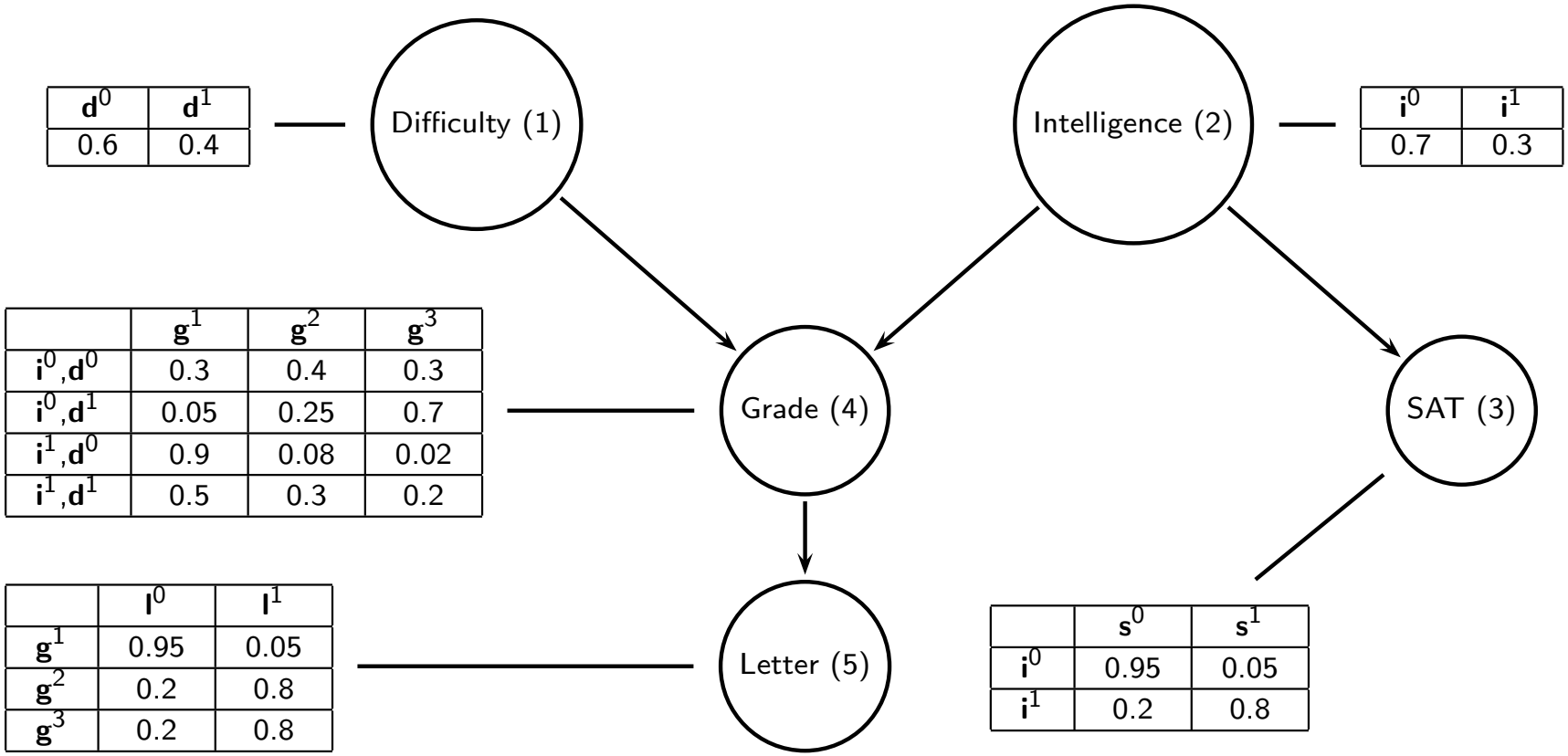
i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

	l^0	l^1
g^1	0.95	0.05
g^2	0.2	0.8
g^3	0.2	0.8

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

```
> C_MMPC(dataSet)
[[Difficulty]]  [[Intelligence]]  [[SAT]]  [[Grade]]  [[Letter]]
[1] 4           [1] 3 4           [1] 2    [1] 5 2 1  [1] 4
```



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

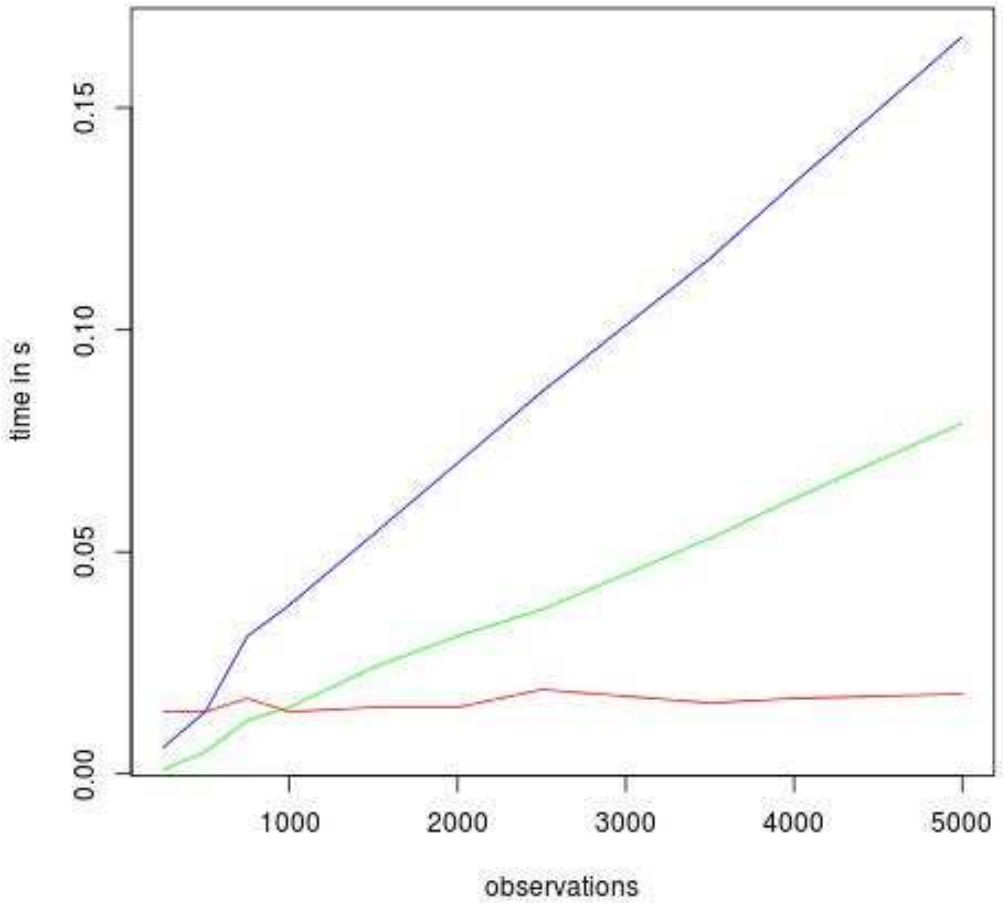
	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

	l^0	l^1
g^1	0.95	0.05
g^2	0.2	0.8
g^3	0.2	0.8

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

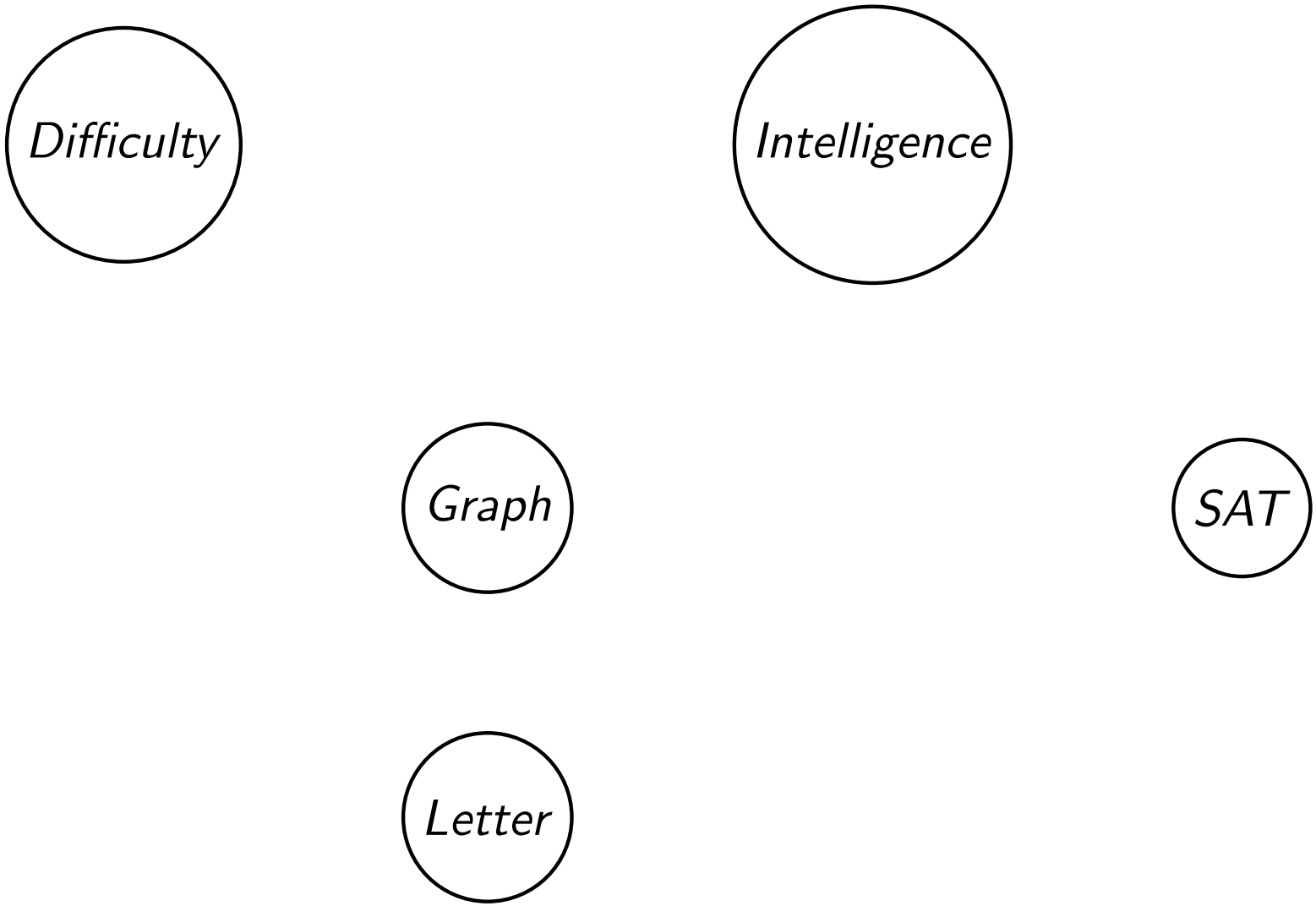
```
> C_MMPC(dataSet)
[[Difficulty]]  [[Intelligence]]  [[SAT]]  [[Grade]]  [[Letter]]
[1] 4           [1] 3 4           [1] 2    [1] 5 2 1  [1] 4
```

The benchmark for this algorithm

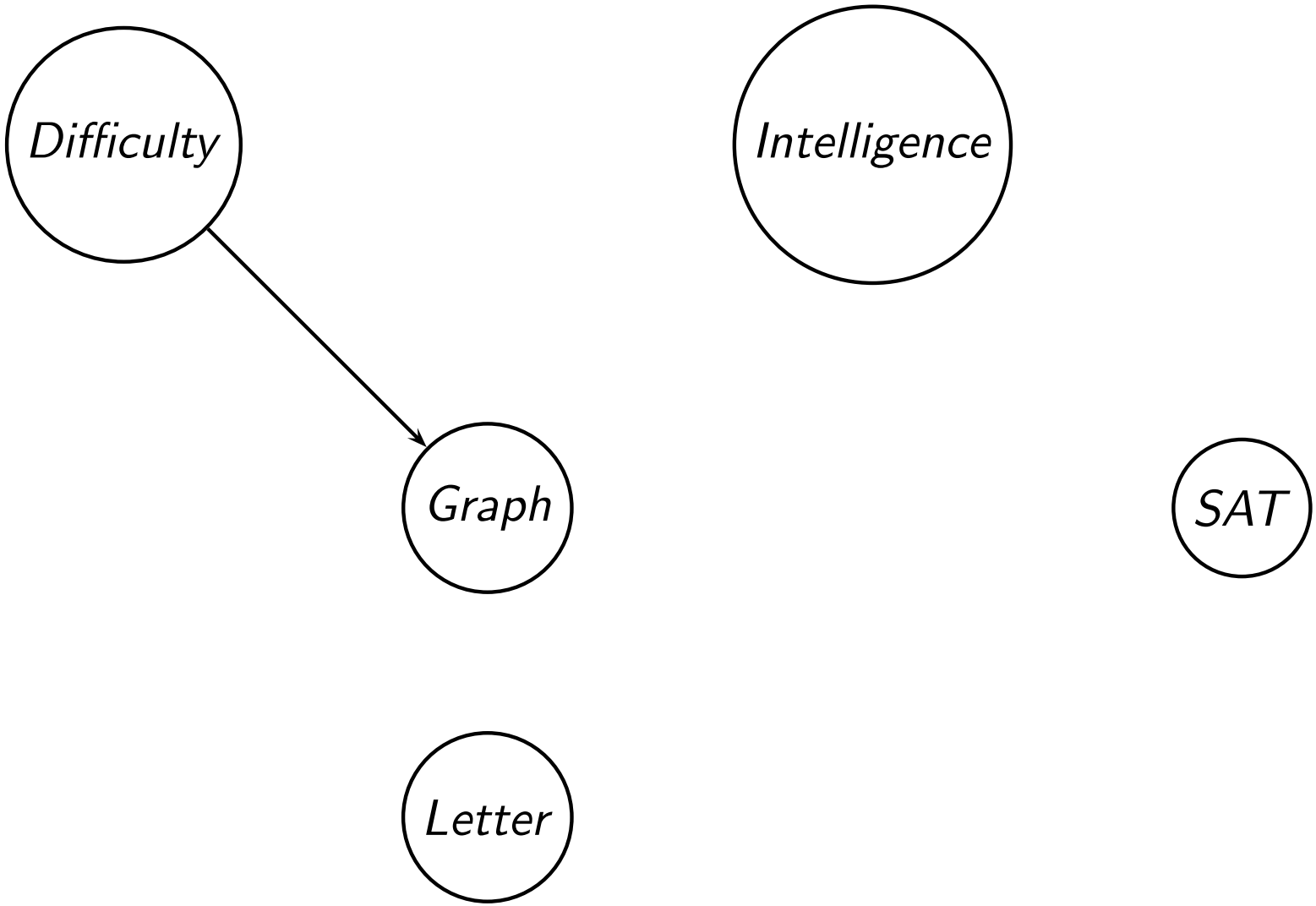


nobs	R	C	bnlearn
250	0.006	0.001	0.014
500	0.014	0.005	0.014
750	0.031	0.012	0.017
1000	0.038	0.015	0.014
1500	0.054	0.024	0.015
2500	0.086	0.037	0.019
5000	0.166	0.079	0.018

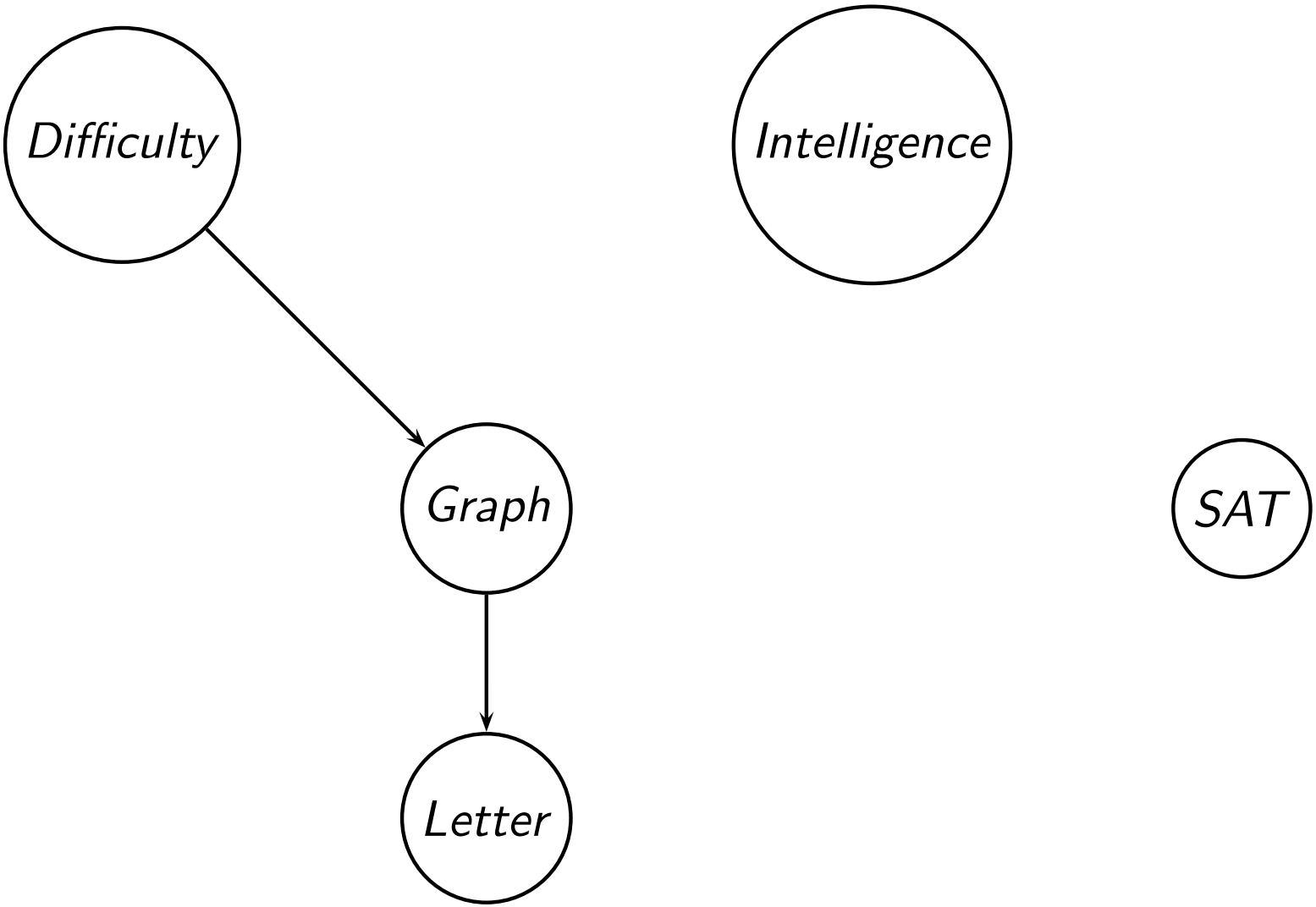
Bayesian Dirichlet equivalent uniform (BDeu) score



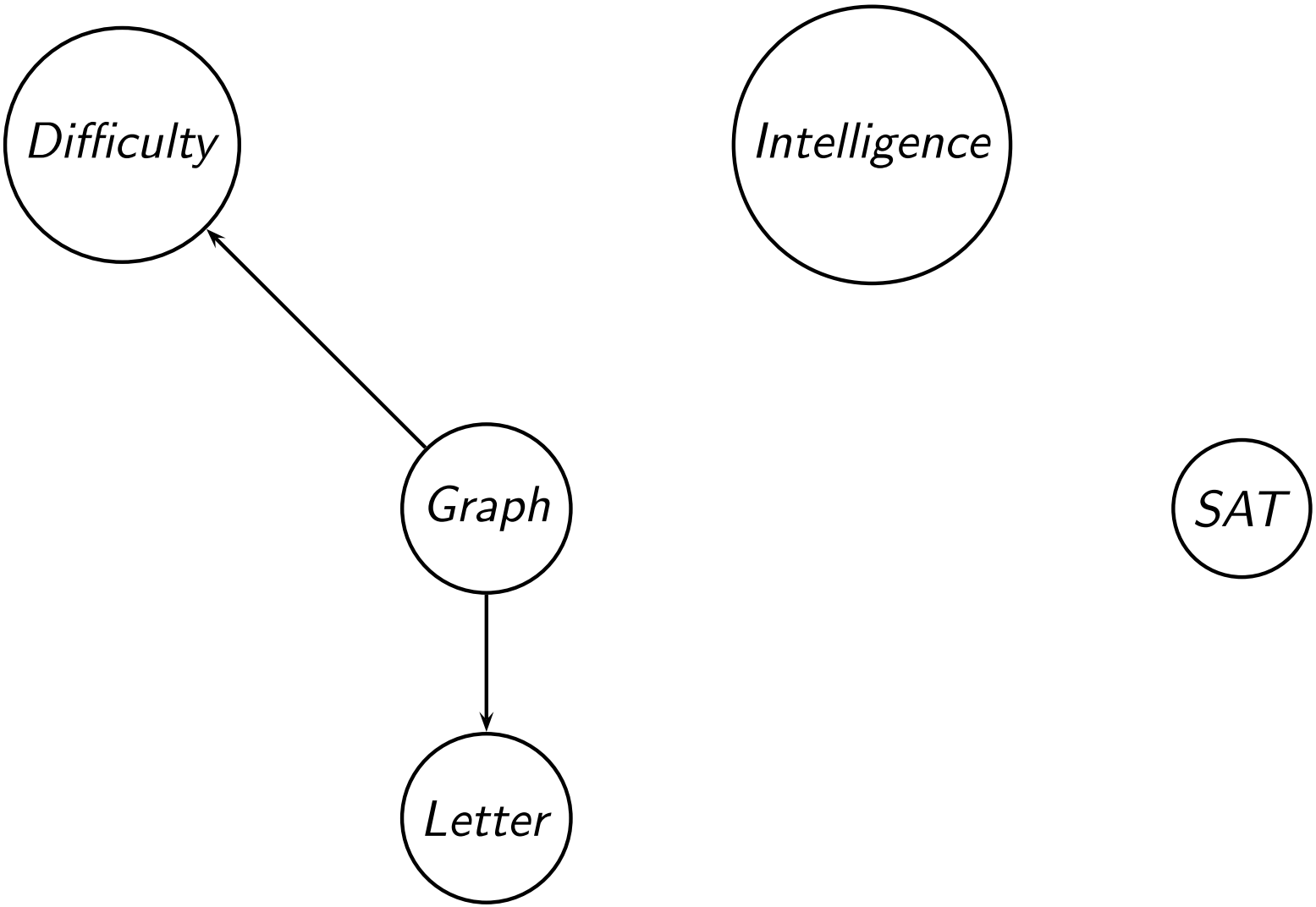
Adding an edge



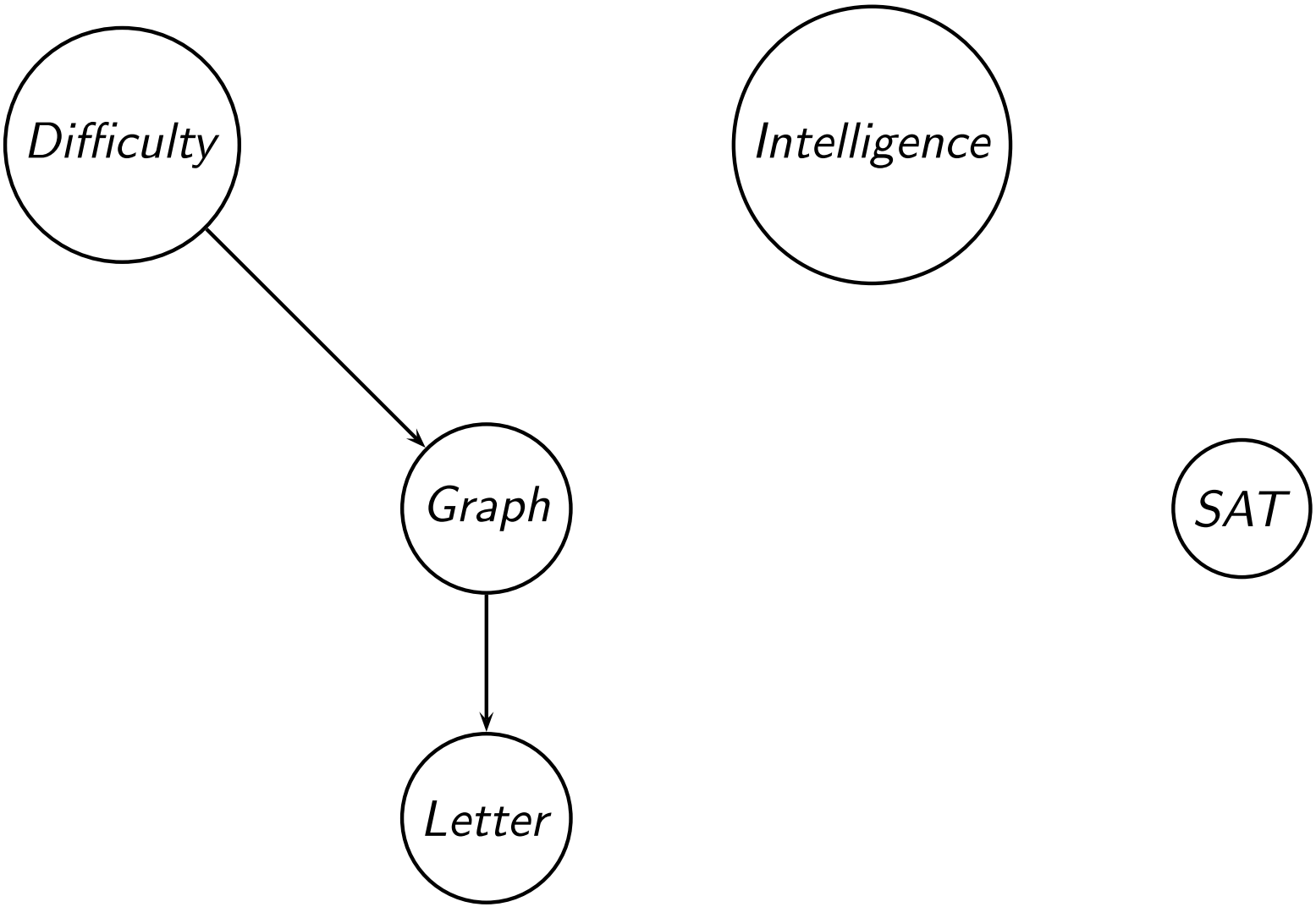
Adding an edge



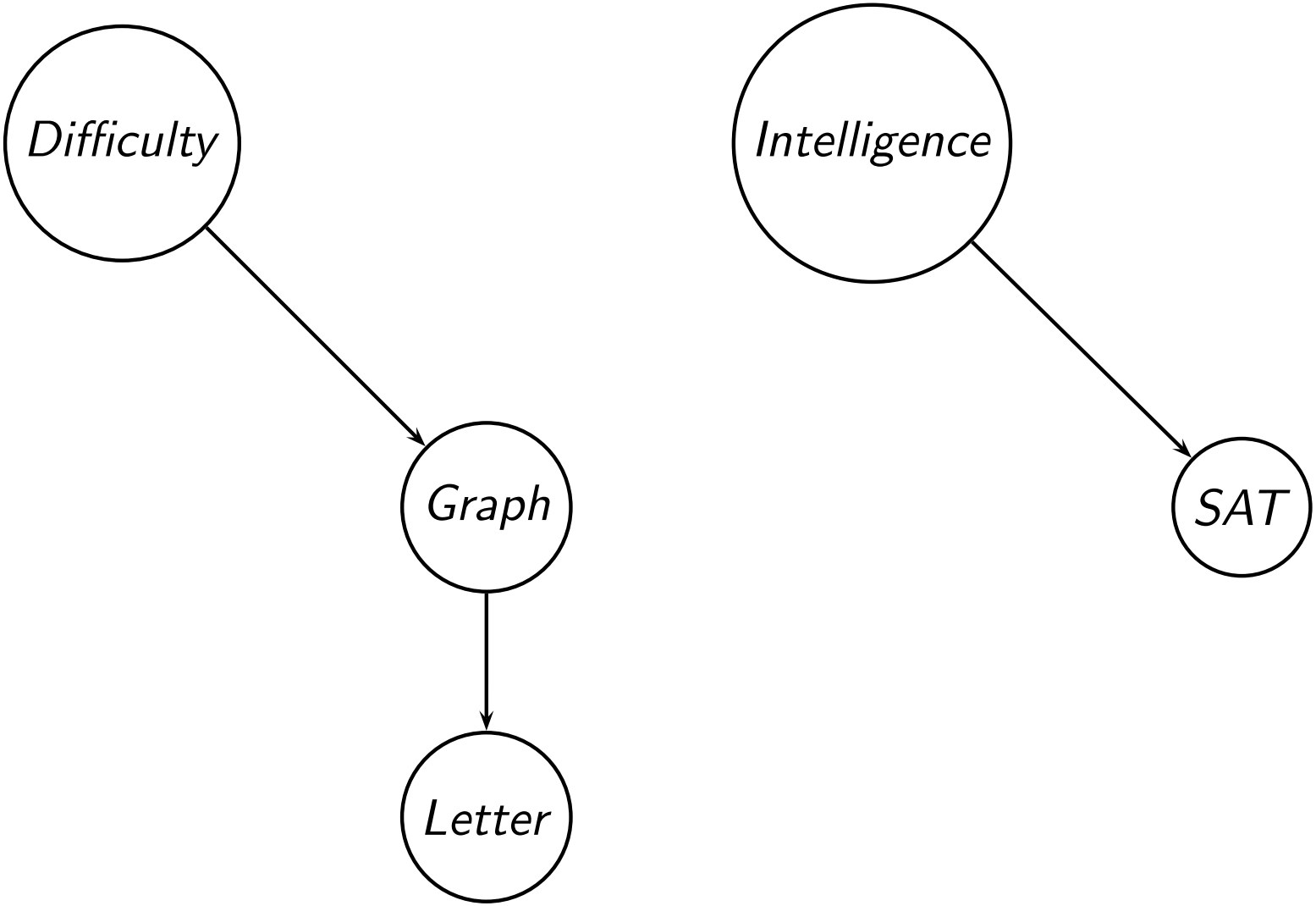
Also possible: reverse and delete



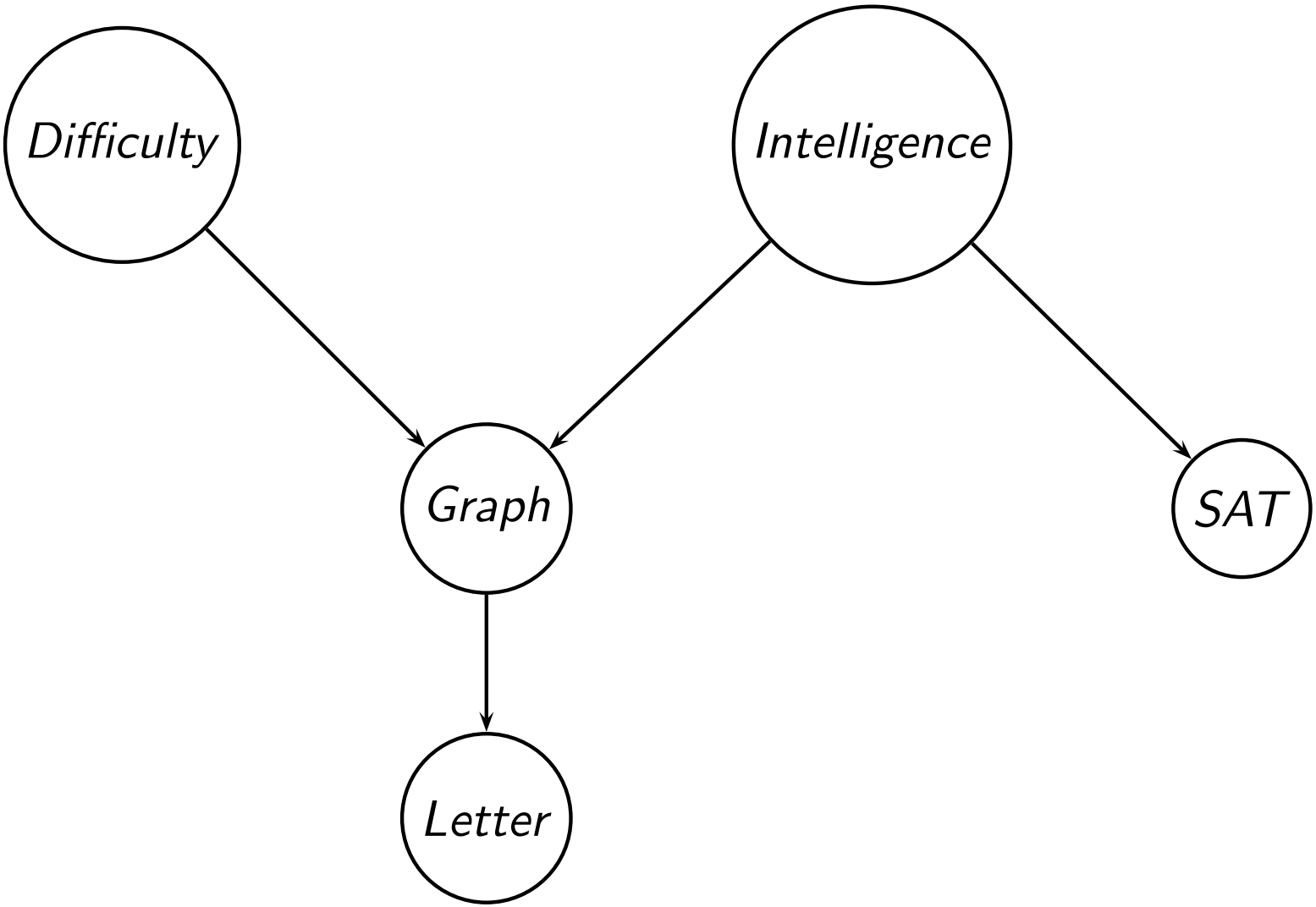
Adding an edge



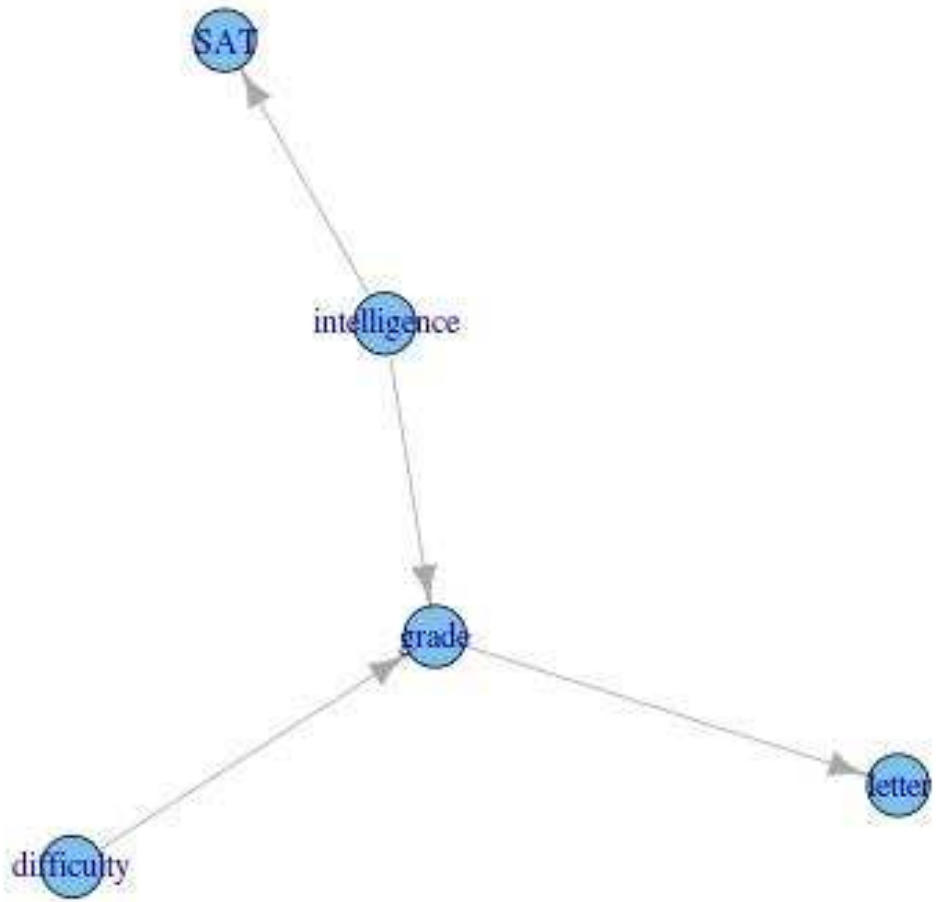
Adding an edge



Adding an edge



Output of my programme



Thanks for your attention!