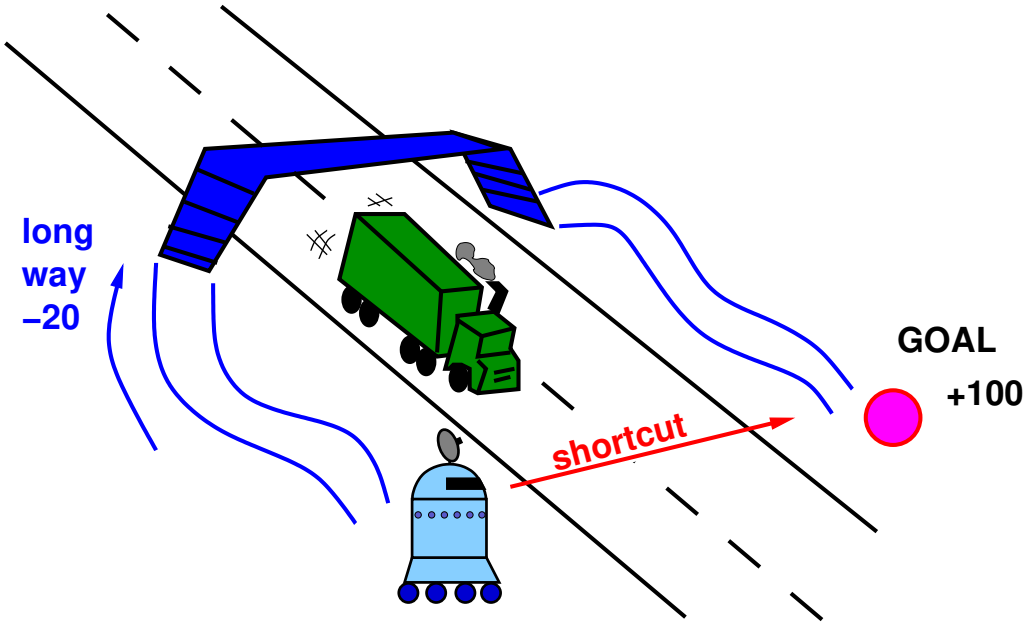
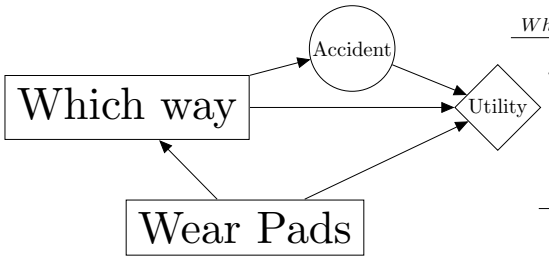
Decision Network Examples

Jesse Hoey
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, CANADA, N2L3G1
jhoey@cs.uwaterloo.ca

To get to its goal, a robot can go one of two ways: a long, safe route and a shortcut. The shortcut crosses a busy road on which the robot gets into an accident with probability 0.2*. The probability of an accident on the long, safe route is* 0.01*. The goal is worth* 100*, but the robot has to pay* 20 *to take the long route. Having an accident costs* 97 *on the short route (leaving the robot with only* 3*) and* 80 *on the long route (leaving the robot with* 0*). However, the robot can put on a set of pads before it starts, but putting on the pads costs* 5*. Getting into an accident while wearing pads only costs* 60 *on the short route (leaving the robot with* 35*) and* 45 *on the long route (leaving the robot with* 30*).*





$WhichWay$	$P(Accident = true \mid WhichWay)$
long	0.01
short	0.2

$WhichWay$	$Accident$	$WearPads$	$U(Accident, WhichWay, WearPads)$
long	*true*	*true*	30
long	*false*	*true*	75
long	*true*	*false*	0
long	*false*	*false*	80
short	*true*	*true*	35
short	*false*	*true*	95
short	*true*	*false*	3
short	*false*	*false*	100

The expected value for the robot is:

$$\mathcal{E}(which\_way, wear\_pads) = \sum_{accident} P(accident|which\_way)U(which\_way, accident, wear\_pads)$$

which can be computed for each possible decision as:

$which\_way$	$wear\_pads$	$\mathcal{E}(which\_way, wear\_pads)$
short	true	0.2*35+0.8*95=83
long	true	0.01*30+0.99*75=74.6
short	false	0.2*3+0.8*100=80.6
long	false	0.01*0+0.99*80=79.2

And so, the optimal decision is to put on the pads and go the short way (with expected value of 83).
We can also compute the minimum probability of accident (p) at which the robot will put on pads by setting the values to be equal. For the short route, this is where

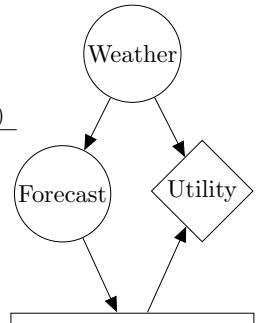
$$p * 35 + (1 - p) * 95 = p * 3 + (1 - p) * 100$$

solving for p gives 0.135. Thus, if $p > 0.135$, the robot will put on the pads if going the short route.
For the long route, call the probabilty of accident q, we get

$$q * 30 + (1 - q) * 75 = q * 0 + (1 - q) * 80$$

gives $q = 0.142$.

When going out, one must decide whether to take an umbrella or not. This decision is not based on the forthcoming weather, but rather on the weather forecast (which is sometimes wrong). Taking an umbrella on a sunny day sucks because it needs to be carried around all day. Not taking an umbrella on a rainy day sucks even more. The network is as shown below.



$Weather$	$P(Weather)$
no_rain	0.7
rain	0.3

$Weather$	$Forecast$	$P(Forecast\|Weather)$
no_rain	*sunny*	0.7
no_rain	*cloudy*	0.2
no_rain	*rainy*	0.1
rain	*sunny*	0.15
rain	*cloudy*	0.25
rain	*rainy*	0.6

$Weather$	$Umbrella$	$U(Weather, Umbrella)$
no_rain	*take*	20
no_rain	*leave*	100
rain	*take*	70
rain	*leave*	0

Factors are:

$$f_0(Weather, Umbrella) = U(Weather, Umbrella)$$
$$f_1(Weather) = P(Weather)$$
$$f_2(Weather, Forecast) = P(Forecast|Weather)$$

Overall, the calculation will be one of

$$\mathcal{E}(Umbrella) = \sum_{Forecast, Weather} P(Weather, Forecast)U(Weather, Umbrella)$$

Using the variable elimination algorithm, we first sum out $Weather$:

$$f_3(Forecast, Umbrella) = \sum_{Weather} f_0(Weather, Umbrella)f_2(Weather, Forecast)f_1(Weather)$$

$=$

$Forecast$	$Umbrella$	$f_3(Forecast, Umbrella)$
sunny	*take*	$0.7 * 0.7 * 20 + 0.3 * 0.15 * 70 = 12.95$
sunny	*leave*	$0.7 * 0.7 * 100 + 0.3 * 0.15 * 0 = 49$
cloudy	*take*	$0.7 * 0.2 * 20 + 0.3 * 0.25 * 70 = 8.05$
cloudy	*leave*	$0.7 * 0.2 * 100 + 0.3 * 0.25 * 0 = 14$
rainy	*take*	$0.7 * 0.1 * 20 + 0.3 * 0.6 * 70 = 14$
rainy	*leave*	$0.7 * 0.1 * 100 + 0.3 * 0.6 * 0 = 7$

Now, we max out $Umbrella$

$$f_4(Forecast) = \max_{Umbrella} f_3(Forecast, Umbrella)$$

$$= \begin{array}{c|cc} Forecast & f_4(Forecast) & policy(Forecast) \\ \hline sunny & 49 & leave \\ cloudy & 14 & leave \\ rainy & 14 & take \end{array}$$

Finally, sum out Forecast to get the $f_5() = 77$ which is the value of the optimal policy: take the umbrella if the forecast is rainy otherwise leave it behind. The number 77 gives the return, on average, that this agent will expect to get from following this policy over a number of days.

Now, suppose we observe $Forecast = cloudy$. We already know the policy is to leave the umbrella, but what is the expected value? Its not 14 as in $f_4(Forecast)$ above, as this is the portion of the overall expected value allocated to situations where $Forecast = cloudy$ (which is not very likely). To answer this query, consider that we have to restrict f_2 to be only over $Forecast = cloudy$.

$Weather$	$f_2(Weather)$
no_rain	0.2
$rain$	0.25

but this is not normalized, so we need to make it a proper probability distribution first, by noting that we are now trying to find

$$\mathcal{E}(Umbrella, Weather) = \sum_{Weather} P(Weather|Forecast = cloudy)U(Weather, Umbrella)$$

so we have to find the factor $f_6(Weather)$ corresponding to

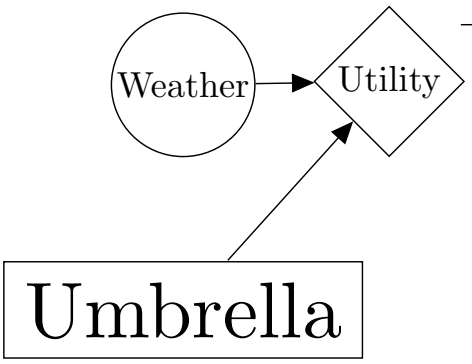
$$P(Weather|Forecast = cloudy) = \frac{P(Forecast = cloudy|Weather)P(Weather)}{P(Forecast)}$$

which is

$Weather$	$f_6(Weather)$
no_rain	$0.2 * 0.7/(0.2 * 0.7 + 0.25 * 0.3) = 0.65$
$rain$	$0.25 * 0.3/(0.2 * 0.7 + 0.25 * 0.3) = 0.35$

What we have done is essentially convert the original decision network into a new one that is restricted for $Forecast = cloudy$, so the variable $Weather$ now has the interpretation of the weather *given* that the forecast is cloudy.

$Weather$	$P(Weather)$
no_rain	0.65
$rain$	0.35



$Weather$	$Umbrella$	$U(Weather, Umbrella)$
no_rain	$take$	20
no_rain	$leave$	100
$rain$	$take$	70
$rain$	$leave$	0

Now we can sum out Weather as before, but with only two factors:

$$f_7(Umbrella) = \sum_{Weather} f_0(Weather, Umbrella) f_6(Weather)$$

	$Umbrella$	$f_3(Forecast, Umbrella)$
=	$take$	$0.65 * 20 + 0.35 * 70 = 37.5$
	$leave$	$0.65 * 100 + 0.35 * 0 = 65$

maxing out $Umbrella$ gives us

$$f_8() = \max_{Umbrella} f_7(Umbrella) = 65$$

and the policy is to leave the umbrella behind and the expected value is 65. This is the actual expected value for all days in which the $Forecast$ was cloudy.
If we don't do the normalization first, and simply run the same variable elminiation algorithm, we get:

$$f_8(Umbrella) = \sum_{Weather} f_0(Weather, Umbrella) f_6(Weather) f_1(Weather)$$

	$Umbrella$	$f_8(Umbrella)$
=	$take$	$0.7 * 0.2 * 20 + 0.3 * 0.25 * 70 = 8.05$
	$leave$	$0.7 * 0.2 * 100 + 0.3 * 0.25 * 0 = 14$

Now, we max out $Umbrella$ to get $f_9() = 14$ and the policy is to leave the umbrella. To get the expected value, we divide by the $P(evidence)$ which is

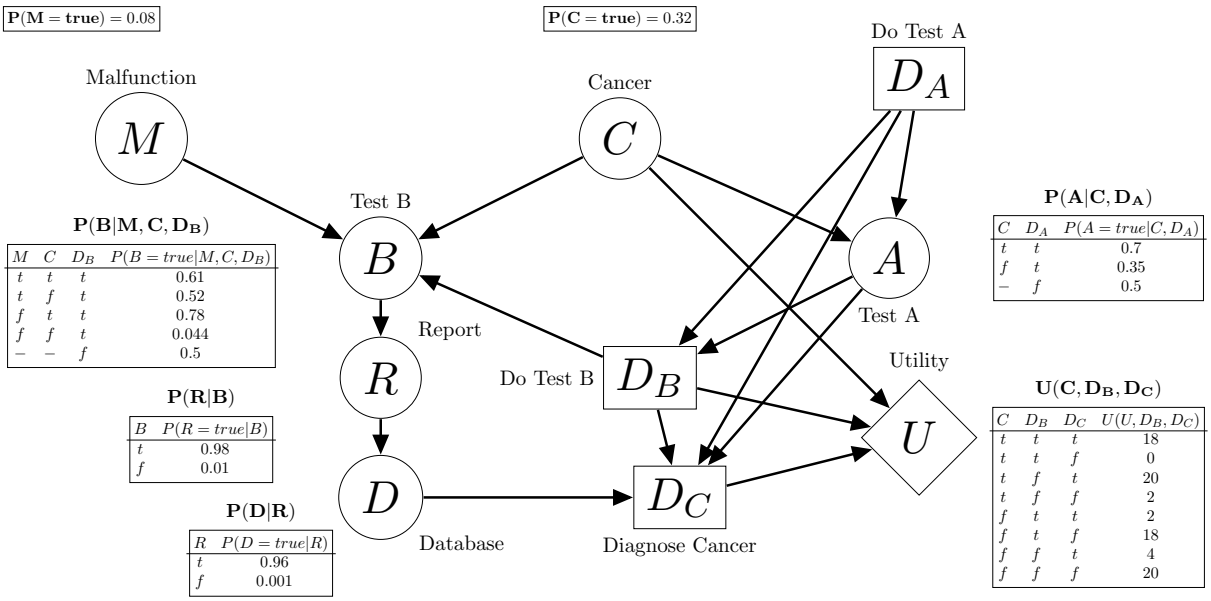
$$P(evidence) = P(Forecast = cloudy)$$
$$= \sum_{Weather} P(Forecast = cloudy|Weather) P(Weather) \quad = 0.2 * 0.7 + 0.25 * 0.3 = 0.215$$

and so $f_{10}() = 14/0.215 = 65$

Cancer is a disease which occurs with probability 0.32 *in a certain population (e.g. older long-time smokers). A Decision network is used to diagnose the presence of cancer (C) in such patients by using two binary tests, test A (A) and test B (B) which report the presence or absence of cancer. Test A is a simple test the doctor can perform directly on the patient and see the results immediately. Test A has a true positive rate of* 0.8 *and a false positive rate of* 0.15*. That is, when cancer is present, test A detects it* 80% *of the time, and when cancer is not present, test A reports it* 15% *of the time. Test A does not cost anything to administer. Test B, on the other hand, has greater precision (true positive rate* 0.78 *and false positive rate* 0.044*), but requires a complicated machine, which malfunctions (M) with probability* 0.08*. When the machine malfunctions, test B's true positive rate drops to* 0.61 *and its false positive rate rises to* 0.52*. Test B costs* 2 *(arbitrary units) to administer. Further, test B's results are not directly available to the doctor. Instead, they are read by a technician who writes them down as a report (R) in a logbook, which is then passed to a data entry person who enters the result in a database (D). The doctor reads the result from the database. The technician and the data entry person sometimes make mistakes, however. The technician's true and false positive rates are* 0.98 *and* 0.01*, respectively, while the data entry person's rates are* 0.96 *and* 0.001 *(for true and false positive rates, respectively).*

The cost function is defined in terms of a maximum reward of 20 *(arbitrary units). The doctor receives* 20 *for diagnosing cancer correctly (whether cancer is actually present or not). Test B costs* 2 *to administer, and a false positive diagnosis (saying cancer is there when it is not) costs* 16*. A false negative diagnosis (saying cancer is not there when it is) costs* 18*.*

The decision network and all conditional probability tables and the utility function are shown here:



We have the following factors:

$f_0(C) = P(C) =$

C	$f_0(C)$
t	0.32
f	0.68

$f_1(M) = P(M) =$

M	$P(M)$
t	0.08
f	0.92

$f_2(B, M, C, D_B) = P(B|M, C, D_B) =$

M	C	B	D_B	$f_1(B, M, C, D_B)$
t	t	t	t	0.61
t	t	f	t	0.39
t	f	t	t	0.52
t	f	f	t	0.48
f	t	t	t	0.78
f	t	f	t	0.22
f	f	t	t	0.044
f	f	f	t	0.956
$-$	$-$	$-$	f	0.5

$f_3(R, B) = P(R|B) =$

R	B	$f_2(R, B)$
t	t	0.98
t	f	0.01
f	t	0.02
f	f	0.99

$f_4(D, R) = P(D|R) =$

D	R	$f_3(D, R)$
t	t	0.96
t	f	0.001
f	t	0.04
f	f	0.999

$f_5(A, C, D_A) = P(A|C, D_A) =$

C	A	D_A	$f_5(C, A, D_A)$
t	t	t	0.8
t	f	t	0.2
f	t	t	0.15
f	f	t	0.85
$-$	$-$	f	0.5

$f_6(C, D_B, D_C) = U(C, D_B, D_C) =$

C	D_B	D_C	$U(U, D_B, D_C)$
t	t	t	18
t	t	f	0
t	f	t	20
t	f	f	2
f	t	t	2
f	t	f	18
f	f	t	4
f	f	f	20

Note that for f_2 and f_5 we have written a short-hand version since if the decision is not made (e.g. if test B or test A is not administered, resp.), then the conditional probability becomes uninformative. The same effect could be achieved by adding a third value *"not done"* to variables A and B, and then having that value have probability 1 when $D_A = false$ or $D_B = false$, respectively. In this case, the tables for R and D would also need to be changed.
Applying variable elimination to this example, we have three decisions to optimize: D_A,D_B and D_C. We first sum out all variables that are not parents of a remaining decision node: R, B, M and C. We get the following sequence of factors:

$$f_7(B, C, D_B) = \sum_M f_1(M) f_2(B, M, C, D_B)$$
$$f_8(R, C, D_B) = \sum_B f_3(R, B) f_7(B, C, D_B)$$
$$f_9(D, C, D_B) = \sum_R f_4(D, R) f_8(R, C, D_B)$$
$$f_{10}(D, A, D_A, D_B, D_C) = \sum_C f_9(D, C, D_B) f_0(C) f_5(A, C, D_A) f_6(C, D_B, D_C)$$

Now, we max out the "last" decision, D_C, giving

$$f_{11}(D, A, D_A, D_B) = \max_{D_C} f_{10}(D, A, D_A, D_B, D_C)$$

We get our first policy here as $\delta_{D_C} = \arg\max_{D_C} f_{10}(D, A, D_A, D_B, D_C)$.
This is a policy for D_C over D, A, D_A, D_B.
Then sum out D since it no longer is a parent of any remaining decision variable, giving

$$f_{12}(A, D_A, D_B) = \sum_D f_{11}(D, A, D_A, D_B)$$

Then, we max out the next decision, D_B, giving

$$f_{13}(A, D_A) = \max_{D_B} f_{12}(A, D_A, D_B)$$

This gives our second policy as $\delta_{D_B} = \arg\max_{D_B} f_{12}(A, D_A, D_B)$.
This is a policy for D_B over A, D_A.
Now we can sum out A, giving

$$f_{14}(D_A) = \sum_A f_{13}(A, D_A)$$

and finally, we max out D_A to get the final value:

$$f_{15}() = \max_{D_A} f_{14}(D_A)$$

and our third policy is $\delta_{D_A} = \arg\max_{D_A} f_{14}(D_A)$. This is a policy for D_A.

The final policies is then:

```
do test A
if Test A=positive,
       Do test B:
           Diagnose according to Database
else (Test A=negative)
       Diagnose no Cancer
```

and the final expected value is 15.9

Here is PostgreSQL code to compute this. I am not a PSQL hacker so this solution is (1) really long and explicit and could be done much more elegantly, I'm sure; and (2) lacking the argmax needed to actually get the policy (at each step with a max), so I just unwind at the end to show, but this gives a more interpretable policy anyways. If you find any errors let me know.

```
CREATE TABLE f0 (C BOOLEAN, P FLOAT);
insert into f0 (C,P) values (True,0.32);
insert into f0 (C,P) values (False,0.68);
select * from f0;
 c |  p
---+------
 t | 0.32
 f | 0.68

CREATE TABLE f1 (M BOOLEAN, P FLOAT);
insert into f1 (M,P) values (True,0.08);
insert into f1 (M,P) values (False,0.92);
 m |  p
---+------
 t | 0.08
 f | 0.92

CREATE TABLE f2 (M BOOLEAN, C BOOLEAN, B BOOLEAN, DB BOOLEAN, P FLOAT);
insert into f2 (M,C,B,DB,P) values (True,True,True,True,0.61);
insert into f2 (M,C,B,DB,P) values (True,True,False,True,0.39);
insert into f2 (M,C,B,DB,P) values (True,False,True,True,0.52);
insert into f2 (M,C,B,DB,P) values (True,False,False,True,0.48);
insert into f2 (M,C,B,DB,P) values (False,True,True,True,0.78);
insert into f2 (M,C,B,DB,P) values (False,True,False,True,0.22);
insert into f2 (M,C,B,DB,P) values (False,False,True,True,0.044);
insert into f2 (M,C,B,DB,P) values (False,False,False,True,0.956);
insert into f2 (M,C,B,DB,P) values (True,True,True,False,0.5);
insert into f2 (M,C,B,DB,P) values (True,True,False,False,0.5);
insert into f2 (M,C,B,DB,P) values (True,False,True,False,0.5);
insert into f2 (M,C,B,DB,P) values (True,False,False,False,0.5);
insert into f2 (M,C,B,DB,P) values (False,True,True,False,0.5);
insert into f2 (M,C,B,DB,P) values (False,True,False,False,0.5);
insert into f2 (M,C,B,DB,P) values (False,False,True,False,0.5);
insert into f2 (M,C,B,DB,P) values (False,False,False,False,0.5);
```

```
  m | c | b | db |   p
---+---+---+----+-------
 t | t | t | t  |  0.61
 t | t | f | t  |  0.39
 t | f | t | t  |  0.52
 t | f | f | t  |  0.48
 f | t | t | t  |  0.78
 f | t | f | t  |  0.22
 f | f | t | t  | 0.044
 f | f | f | t  | 0.956
 t | t | t | f  |   0.5
 t | t | f | f  |   0.5
 t | f | t | f  |   0.5
 t | f | f | f  |   0.5
 f | t | t | f  |   0.5
 f | t | f | f  |   0.5
 f | f | t | f  |   0.5
 f | f | f | f  |   0.5
```

```
CREATE TABLE f3 (R BOOLEAN, B BOOLEAN, P FLOAT);
insert into f3 (R,B,P) values (True,True,0.98);
insert into f3 (R,B,P) values (True,False,0.01);
insert into f3 (R,B,P) values (False,True,0.02);
insert into f3 (R,B,P) values (False,False,0.99);
```

```
 r | b |  p
---+---+------
 t | t | 0.98
 t | f | 0.01
 f | t | 0.02
 f | f | 0.99
```

```
CREATE TABLE f4 (D BOOLEAN, R BOOLEAN, P FLOAT);
insert into f4 (D,R,P) values (True,True,0.96);
insert into f4 (D,R,P) values (True,False,0.001);
insert into f4 (D,R,P) values (False,True,0.04);
insert into f4 (D,R,P) values (False,False,0.999);
```

```
 d | r |   p
---+---+-------
 t | t |  0.96
 t | f | 0.001
 f | t |  0.04
 f | f | 0.999
```

```
CREATE TABLE f5 (A BOOLEAN, C BOOLEAN, DA BOOLEAN, P FLOAT);
insert into f5 (A,C,DA,P) values (True,True,True,0.7);
insert into f5 (A,C,DA,P) values (False,True,True,0.3);
insert into f5 (A,C,DA,P) values (True,False,True,0.35);
insert into f5 (A,C,DA,P) values (False,False,True,0.65);
insert into f5 (A,C,DA,P) values (True,True,False,0.5);
insert into f5 (A,C,DA,P) values (False,True,False,0.5);
insert into f5 (A,C,DA,P) values (True,False,False,0.5);
insert into f5 (A,C,DA,P) values (False,False,False,0.5);
```

a	c	da	p
t	t	t	0.7
f	t	t	0.3
t	f	t	0.35
f	f	t	0.65
t	t	f	0.5
f	t	f	0.5
t	f	f	0.5
f	f	f	0.5

```
CREATE TABLE f6 (C BOOLEAN, DB BOOLEAN, DC BOOLEAN, P FLOAT);
insert into f6 (C,DB,DC,P) values (True,True,True,18);
insert into f6 (C,DB,DC,P) values (True,True,False,0);
insert into f6 (C,DB,DC,P) values (True,False,True,20);
insert into f6 (C,DB,DC,P) values (True,False,False,2);
insert into f6 (C,DB,DC,P) values (False,True,True,2);
insert into f6 (C,DB,DC,P) values (False,True,False,18);
insert into f6 (C,DB,DC,P) values (False,False,True,4);
insert into f6 (C,DB,DC,P) values (False,False,False,20);
```

c	db	dc	p
t	t	t	18
t	t	f	0
t	f	t	20
t	f	f	2
f	t	t	2
f	t	f	18
f	f	t	4
f	f	f	20

Now, start the computations

```
create table f7 as (select C,B,DB,SUM(P) as P from
         (select f1.m,c,b,db,f2.p*f1.p as p from f2,f1 where f1.m=f2.m)
         as subquery group by C,B,DB);
```

```
c | b | db |          p
---+---+----+---------------------
 f | t | f  |                 0.5
 f | f | t  |             0.91792
 t | t | f  |                 0.5
 t | f | t  |              0.2336
 t | f | f  |                 0.5
 f | t | t  | 0.08208000000000001
 f | f | f  |                 0.5
 t | t | t  |              0.7664
```

```
create table f8 as (select R,C,DB,SUM(P) as P from
          (select f3.b,r,c,db,f3.p*f7.p as p from f3,f7 where f3.b=f7.b)
          as subquery group by R,C,DB);
```

```
 r | c | db |         p
---+---+----+--------------------
 f | t | f  |             0.505
 f | f | t  | 0.9103823999999999
 t | t | f  |             0.495
 t | f | t  |          0.0896176
 t | f | f  |             0.495
 f | t | t  |           0.246592
 f | f | f  |             0.505
 t | t | t  |           0.753408
```

```
create table f9 as (select D,C,DB,SUM(P) as P from
         (select f4.r,d,c,db,f4.p*f8.p as p from f4,f8 where f4.r=f8.r)
         as subquery group by D,C,DB);
```

```
d | c | db |          p
---+---+----+---------------------
 f | t | f  |  0.5242950000000001
 f | f | t  |        0.9130567216
 t | t | f  | 0.47570499999999993
 t | f | t  |        0.0869432784
 t | f | f  | 0.47570499999999993
 f | t | t  | 0.27648172800000004
 f | f | f  |  0.5242950000000001
 t | t | t  |         0.723518272
```

```
create table f10 as (select D,A,DA,DB,DC,SUM(P) as P from
          (select f9.C,f9.D,A,DA,f9.DB,DC,f9.p*f0.p*f5.p*f6.p as p
                  from f9,f0,f5,f6 where f9.c=f0.c AND f9.C=f5.C AND
                                         f9.C=f6.C AND f0.c=f5.c AND
                                         f0.c=F6.C and f5.C=f6.C and f9.DB=f6.DB)
          as subquery group by D,A,DA,DB,DC);
```

d	a	da	db	dc	p
f	f	t	t	t	1.2849025678784
f	f	f	t	f	5.587907136192
t	t	f	t	f	0.532092863808
f	t	f	t	t	1.4171459473280001
t	f	t	f	t	1.7544000399999997
t	f	f	t	t	2.142854052672
t	t	t	t	f	0.3724650046656
f	f	t	t	f	7.2642792770496
f	f	t	f	f	4.73543244
f	t	f	f	t	2.3907852000000003
f	t	f	f	f	3.7329804000000006
t	t	t	t	t	2.9586106732224
t	f	t	t	f	0.6917207229504001
f	t	t	t	t	1.5493893267776
t	f	f	t	f	0.532092863808
t	t	f	f	t	2.1692147999999998
f	f	f	f	f	3.7329804000000006
f	f	f	f	t	2.3907852000000003
f	f	t	f	t	1.9335999600000005
t	f	f	f	t	2.1692147999999998
f	t	t	t	f	3.9115349953344
t	t	f	t	t	2.142854052672
f	t	t	f	f	2.7305283600000005
t	t	t	f	t	2.58402956
t	t	t	f	f	2.4774716399999996
t	f	f	f	f	3.3870196
t	f	t	t	t	1.3270974321215998
t	f	t	f	f	4.29656756
t	t	f	f	f	3.3870196
f	t	t	f	t	2.84797044
f	f	f	t	t	1.4171459473280001
f	t	f	t	f	5.587907136192

```
create table f11 as (select D,A,DA,DB,max(P) as P from f10 group by D,A,DA,DB);
```

```
 d | a | da | db |         p
---+---+----+----+--------------------
 f | f | t  | t  |    7.2642792770496
 t | f | t  | t  | 1.3270974321215998
 t | t | f  | t  |     2.142854052672
 t | f | f  | t  |     2.142854052672
 f | f | t  | f  |         4.73543244
 f | t | f  | t  |     5.587907136192
 f | t | t  | t  |    3.9115349953344
 t | f | t  | f  |         4.29656756
 f | t | f  | f  | 3.7329804000000006
 f | t | t  | f  |         2.84797044
 t | t | f  | f  |          3.3870196
 t | t | t  | t  |    2.9586106732224
 t | t | t  | f  |         2.58402956
 f | f | f  | f  | 3.7329804000000006
 t | f | f  | f  |          3.3870196
 f | f | f  | t  |     5.587907136192
```

```
create table f12 as (select A,DA,DB,SUM(P) as P from f11 group by A,DA,DB);
```

```
 a | da | db |         p
---+----+----+-------------------
 f | t  | f  |             9.032
 t | t  | f  |             5.432
 f | f  | t  |    7.730761188864
 t | f  | t  |    7.730761188864
 t | f  | f  | 7.120000000000001
 f | t  | t  |   8.5913767091712
 f | f  | f  | 7.120000000000001
 t | t  | t  |   6.8701456685568
```

```
create table f13 as (select A,DA,max(P) as P from f12 group by A,DA);
```

```
 a | da |        p
---+----+-----------------
 f | f  |  7.730761188864
 t | t  | 6.8701456685568
 t | f  |  7.730761188864
 f | t  |           9.032
```

```
create table f14 as (select DA,SUM(P) as P from f13 group by DA);

 da |        p
----+------------------
 f  |  15.461522377728
 t  | 15.9021456685568



create table f15 as (select MAX(P) as P from f14);

       p
------------------
 15.9021456685568



## see policies
select * from f14;
da |        p
----+------------------
 f  |  15.461522377728
 t  | 15.9021456685568
policy: da=t

select a,db,p from f12 where da=true;
 a | db |        p
---+----+-----------------
 f | f  |           9.032
 t | f  |           5.432
 f | t  | 8.5913767091712
 t | t  | 6.8701456685568

policy: if a=t, db=t; if a=f db=f
```

```
select d,dc,p from f10 where da=true and a=true and db=true;
d | dc |        p
---+----+-----------------
 t | f  | 0.3724650046656
 t | t  | 2.9586106732224
 f | t  | 1.5493893267776
 f | f  | 3.9115349953344
policy: if d=t, dc=t; if d=f, dc=f;

select d,dc,p from f10 where da=true and a=false and db=false
 d | dc |         p
---+----+--------------------
 t | t  | 1.7544000399999997
 f | f  |         4.73543244
 f | t  | 1.9335999600000005
 t | f  |         4.29656756
policy: dc=f
```