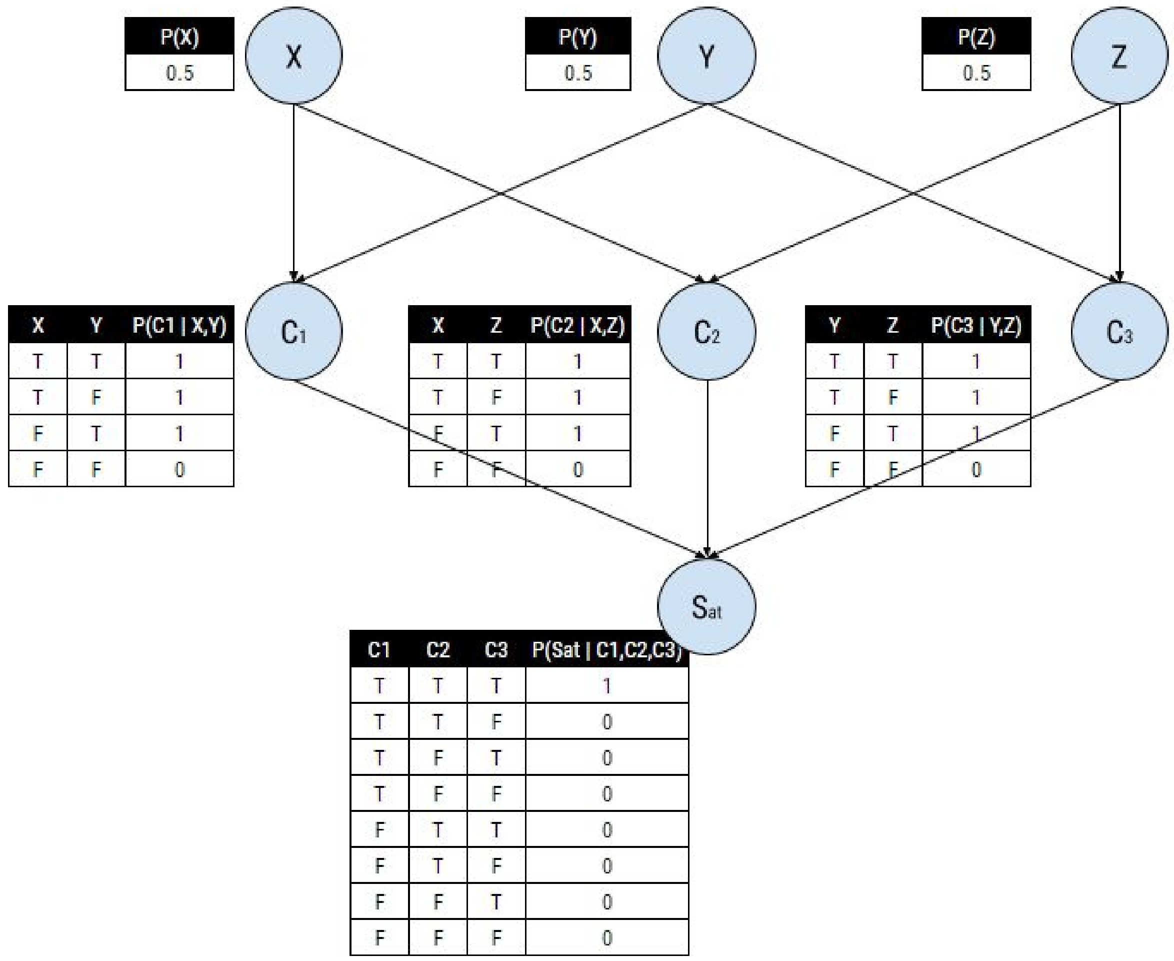
14.16 Investigate the complexity of exact inference in general Bayesian networks:
a. Prove that any 3-SAT problem can be reduced to exact inference in a Bayesian network constructed to represent the particular problem and hence that exact inference is NP-hard. (Hint: Consider a network with one variable for each proposition symbol, one for each clause, and one for the conjunction of clauses.)
b. The problem of counting the number of satisfying assignments for a 3-SAT problem is #P-complete. Show that exact inference is at least as hard as this.

a. Prove that any 3-SAT problem can be reduced to exact inference in a Bayesian network constructed to represent the particular problem and hence that exact inference is NP-hard. (Hint: Consider a network with one variable for each proposition symbol, one for each clause, and one for the conjunction of clauses.)

First suppose there is a 2-SAT problem with 3 propositions X, Y and Z.

Lets try with a problem that presents 3 possible two-literals clauses represented here as nodes C1, C2 and C3: $(X \lor Y) \land (X \lor Z) \land (Y \lor Z)$



P(X)
0.5

P(Y)
0.5

P(Z)
0.5

X	Y	P(C1 | X,Y)
T	T	1
T	F	1
F	T	1
F	F	0

X	Z	P(C2 | X,Z)
T	T	1
T	F	1
F	T	1
F	F	0

Y	Z	P(C3 | Y,Z)
T	T	1
T	F	1
F	T	1
F	F	0

C1	C2	C3	P(Sat | C1,C2,C3)
T	T	T	1
T	T	F	0
T	F	T	0
T	F	F	0
F	T	T	0
F	T	F	0
F	F	T	0
F	F	F	0

The number of clauses in a problem could be lower or greater. There could be i.e. clauses with 2 literals but 1 proposition like (X v X) or maybe (X v -X).

Now, if none evidence is given we can check if problem is solvable when this probability is greater than zero:

$$P(S_{at} = True) = \frac{\sum_x \sum_y \sum_z P(S_{at}=True, C_1=True, C_2=True, C_3=True, x, y, z)}{\sum_{s_{at}} \sum_{c_1} \sum_{c_2} \sum_{c_3} \sum_x \sum_y \sum_z P(s_{at}, c_1, c_2, c_3, x, y, z)}$$

$$P(S_{at} = True) \ = \sum_x \sum_y \sum_z P(S_{at} = True,\ C_1 = True, C_2 = True, C_3 = True,\ x,\ y,\ z)$$

$$P(Sat = T) = P(S_{at} = T \mid C_1 = T,\ C_2 = T,\ C_3 = T) \times \sum_x \sum_y P(C_1 = T \mid x,y) \times \sum_z P(C_2 = T \mid x,z) \times P(C_3 = T \mid y,z) \times P(x) \times P(y) \times P(z)$$

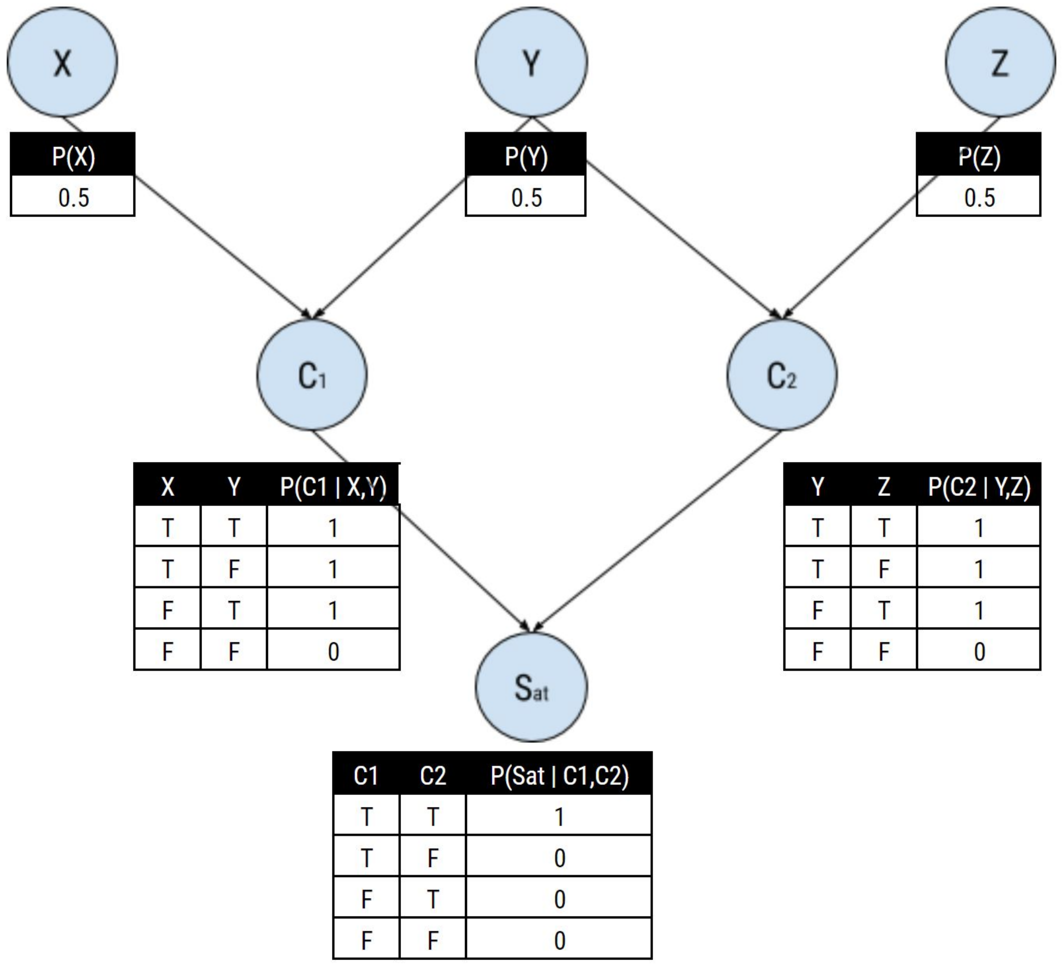
$$P(Sat = True) = \sum_x \sum_y P(C_1 = True \mid x,y) \times \sum_z P(C_2 = True \mid x,z) \times P(C_3 = True \mid y,z) \times P(x) \times P(y) \times P(z)$$

Sat	C1	C2	C3	X	Y	Z	P(Sat | C1,C2,C3)	P(C1 | X,Y)	P(C2 | X,Z)	P(C3 | Y,Z)	P(X)	P(Y)	P(Z)	P
T	T	T	T	T	T	T	1	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	T	T	F	1	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	T	F	T	1	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	T	F	F	1	1	1	0	0.5	0.5	0.5	0
T	T	T	T	F	T	T	1	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	F	T	F	1	1	0	1	0.5	0.5	0.5	0
T	T	T	T	F	F	T	1	0	1	1	0.5	0.5	0.5	0
T	T	T	T	F	F	F	1	0	0	0	0.5	0.5	0.5	0
														0.5

In the above formula and table it is easy to see that Sat will be greater than zero when all clauses are True, and it is derived that the number of cases that will be evaluated by the inference algorithms are directly related with the number of proposition in the network. There are a total of 3 propositions, then there are evaluated $2^3 = 8$ cases. $P(S_{at} = True) = 0.5$

Now lets try with a problem that presents 2 possible two-literals clauses represented here as nodes C1 and C2: $(X \vee Y) \wedge (Y \vee Z)$

Sat	C1	C2	X	Y	Z	P(Sat | C1,C2)	P(C1 | X,Y)	P(C2 | Y,Z)	P(X)	P(Y)	P(Z)	P
T	T	T	T	T	T	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	T	F	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	F	T	1	1	1	0.5	0.5	0.5	0.125
T	T	T	T	F	F	1	1	0	0.5	0.5	0.5	0
T	T	T	F	T	T	1	1	1	0.5	0.5	0.5	0.125
T	T	T	F	T	F	1	1	1	0.5	0.5	0.5	0.125
T	T	T	F	F	T	1	0	1	0.5	0.5	0.5	0
T	T	T	F	F	F	1	0	0	0.5	0.5	0.5	0
												0.625



P(X)
0.5

P(Y)
0.5

P(Z)
0.5

X	Y	P(C1 | X,Y)
T	T	1
T	F	1
F	T	1
F	F	0

Y	Z	P(C2 | Y,Z)
T	T	1
T	F	1
F	T	1
F	F	0

C1	C2	P(Sat | C1,C2)
T	T	1
T	F	0
F	T	0
F	F	0

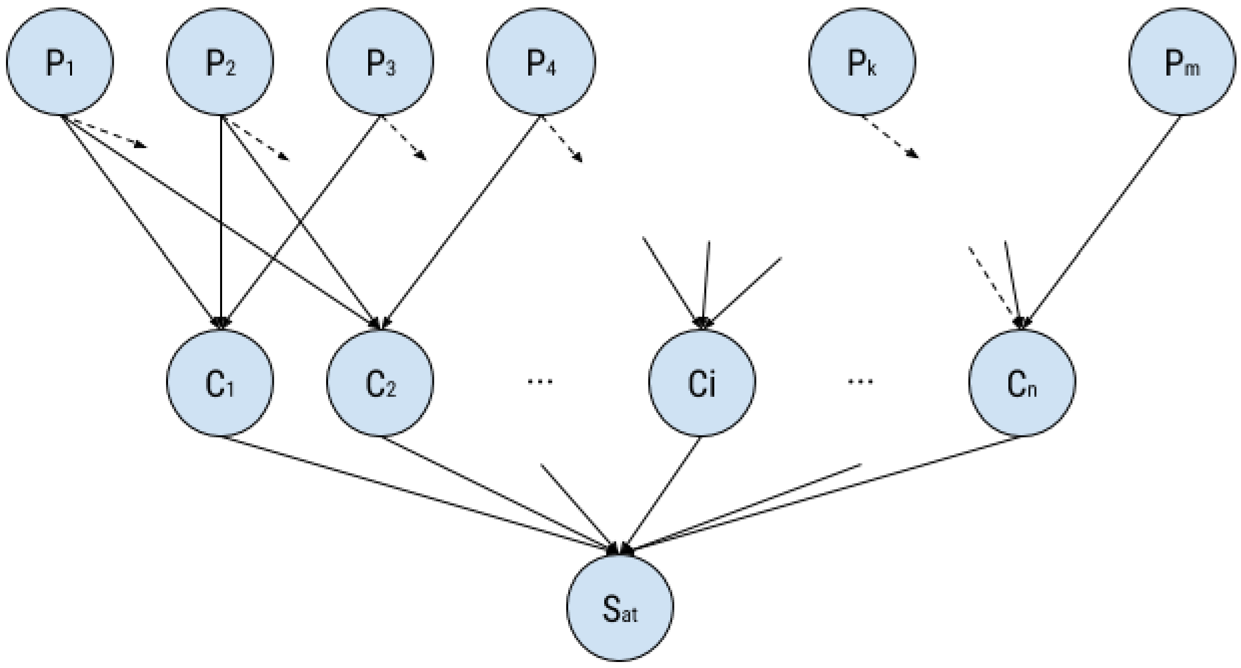
Again we can see that the number of cases evaluated is directly related with the number of proposition and not with the number of clauses. Evaluated cases: $2^3 = 8$. Probability that Sat is True is greater than zero, then the problem is solvable. $P(S_{at} = True) = 0.625$

Building the Bayesian network of this exercise:

We analyze the problem clauses. We start by extracting proposition from clauses and then we add them to the network with no parent with a probability of being True equal to 0.5 in its conditional probability table. Then, we add the clauses one by one with its parents (two propositions already in the network). Here the clause CPT will save a maximum of 4 values: P(Clause_i=True | Pj, Pk). It could be possible to find a clause using only one proposition (Pj v Pj) for which will be required to save only 2 values. After that, we will add the node for variable Sat that will present all the clauses as its parents. A real Bayesian Network will use a CPT related directly to the number of clauses nodes already present in the network. By will could hack this CPT and say that when all clauses are True, the probability of Sat is 1, and 0 otherwise.

- There are a total of m propositions nodes denoted as P_k $with\ 1 \leq k \leq m\ and\ m \in N$
- Nodes P_k represent a total of m boolean variables with no parents at all.
- There are a total of n unique clauses C_i that are children nodes of some P_k combinations.

- Every variable C_i presents a total of 3 P_k parent nodes.
- Variable S_{at} presents all clauses nodes as parents.



b. The problem of counting the number of satisfying assignments for a 3-SAT problem is #P-complete. Show that exact inference is at least as hard as this.

$$P(S_{at} = True) = \sum_{p_1}\sum_{p_2}\sum_{p_3} P(C_1 = True \mid p_1, p_2, p_3) \times \sum_{p_4} P(C_2 = True \mid p_1, p_2, p_4) \times \ldots \times \sum_{p_m} P(C_n = True \mid p_{m-2}, p_{m-1}, p_m)$$

Number of cases evaluated $= 2^m$

$$P(S_{at} = True) = n_{solutions} \times 2^{-m}$$

Where $n_{solutions}$ is the number of assignments to propositions $(P_1,\ P_2, \ldots,\ P_m)$ that makes $P(S_{at} = True \mid P_1,\ P_2, \ldots,\ P_m) > 0$