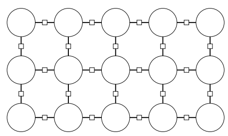
Markov networks: conditional independence

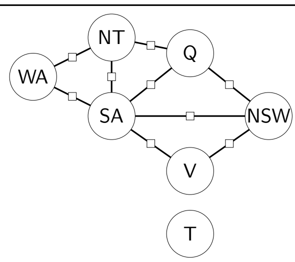


- In this module, I will talk about conditional independence, which allows us to connect the probabilistic notion of independence with connectivity properties of the underlying factor graph.

Motivation

Key idea: graph

Leverage graph properties to derive efficient algorithms which are exact.

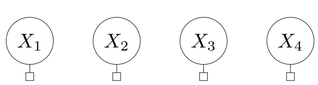


- The goal is to take advantage of the fact that we have a factor **graph**. We will see how exploiting the graph properties can lead us to more efficient algorithms as well as a deeper understanding of the structure of our problem.

Motivation

Backtracking search:

exponential time in number of variables n



Efficient algorithm:

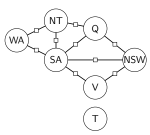
maximize each variable separately

- Recall that backtracking search takes time exponential in the number of variables n. While various heuristics can have dramatic speedups in practice, it is not clear how to characterize those improvements rigorously.
- As a motivating example, consider a fully disconnected factor graph. (Imagine n people trying to vote red or blue, but they don't talk to each other.) It's clear that to get the maximum weight assignment, we can just choose the value of each variable that maximizes its own unary factor without worrying about other variables.

Independence

Definition: independence

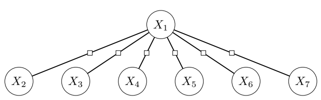
- Let A and B be a partitioning of variables X.
- We say A and B are **independent** if there are no edges between A and B.
- In symbols: $A \perp\!\!\!\perp B$.



$\{$WA, NT, SA, Q, NSW, V$\}$ and $\{$T$\}$ are independent.

- Let us formalize this intuition with the notion of **independence**. It turns out that this notion of independence is deeply related to the notion of independence in probability, as we will see in due time.
- Note that we are defining independence purely in terms of the graph structure, which will be important later once we start operating on the graph using two transformations: conditioning and elimination.

Non-independence

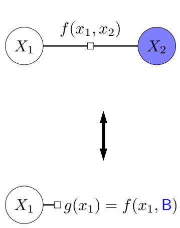


No variables are independent of each other, but feels close...

- When all the variables are independent, finding the maximum weight assignment is easily solvable in time linear in n, the number of variables. However, this is not a very interesting factor graph, because the whole point of a factor graph is to model dependencies (preferences and constraints) between variables.
- Consider the tree-structured factor graph, which corresponds to $n-1$ people talking only through a leader. Nothing is independent here, but intuitively, this graph should be pretty close to independent.

Conditioning

Goal: try to disconnect the graph



x_1	x_2	$f(x_1, x_2)$
R	R	1
R	B	7
B	R	3
B	B	2

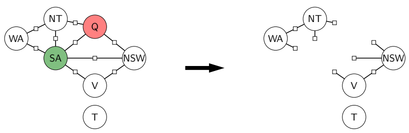
x_1	$g(x_1)$
R	7
B	2

Condition on $X_2 = \text{B}$: remove X_2, f and add g

Conditioning: example

Example: map coloring

Condition on Q = R and SA = G.



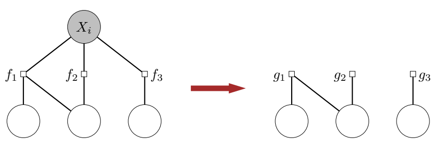
New factors:

[NT ≠ R]
[NSW ≠ R]

[WA ≠ G]
[NT ≠ G]
[NSW ≠ G]
[V ≠ G]

Conditioning: general

Graphically: remove edges from X_i to dependent factors



Definition: conditioning

- To **condition** on a variable $X_i = v$, consider all factors $f_1, \dots, f_k$ that depend on X_i.
- Remove X_i and $f_1, \dots, f_k$.
- Add $g_j(x) = f_j(x \cup \{X_i : v\})$ for $j = 1, \dots, k$.

- In general, factor graphs are not going to have many partitions which are independent (we got lucky with Tasmania, Australia). But perhaps we can transform the graph to make variables independent. This is the idea of **conditioning**: when we condition on a variable $X_i = v$, this is simply saying that we're just going to clamp the value of X_i to v.
- We can understand conditioning in terms of a graph transformation. For each factor f_j that depends on X_i, we create a new factor g_j. The new factor depends on the scope of f_j excluding X_i; when called on x, it just invokes f_j with $x \cup \{X_i : v\}$. Think of g_j as a partial evaluation of f_j in functional programming. The transformed factor graph will have each g_j in place of the f_j and also not have X_i.

Conditional independence

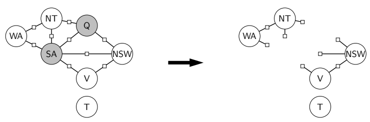
Definition: conditional independence

- Let A, B, C be a partitioning of the variables.
- We say A and B are **conditionally independent** given C if conditioning on C produces a graph in which A and B are independent.
- In symbols: $A \perp\!\!\!\perp B \mid C$.

Equivalently: every path from A to B goes through C.

Conditional independence

Example: map coloring



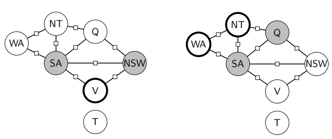
Conditional independence assertion:

$\{\text{WA}, \text{NT}\} \perp\!\!\!\perp \{\text{V}, \text{NSW}, \text{T}\} \mid \{\text{SA}, \text{Q}\}$

- With conditioning in hand, we can define **conditional independence**, perhaps the most important property in factor graphs.
- Graphically, if we can find a subset of the variables $C \subset X$ that disconnects the rest of the variables into A and B, then we say that A and B are conditionally independent given C.
- Later, we'll see how this definition relates to the definition of conditional independence in probability.

Markov blanket

How can we separate an arbitrary set of nodes from everything else?

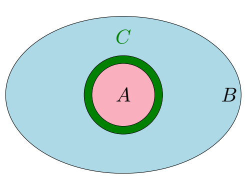


Definition: Markov blanket

Let $A \subseteq X$ be a subset of variables.

Define MarkovBlanket(A) be the neighbors of A that are not in A.

Markov blanket



Proposition: conditional independence

Let $C =$ MarkovBlanket(A).
Let B be $X \backslash (A \cup C)$.
Then $A \perp\!\!\!\perp B \mid C$.

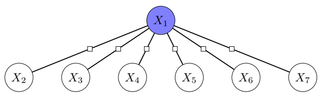
- Suppose we wanted to disconnect a subset of variables $A \subset X$ from the rest of the graph. What is the smallest set of variables C that we need to condition on to make A and the rest of the graph ($B = X \backslash (A \cup C)$) conditionally independent.
- It's intuitive that the answer is simply all the neighbors of A (those that share a common factor) which are not in A. This concept is useful enough that it has a special name: **Markov blanket**.
- Intuitively, the smaller the Markov blanket, the easier the factor graph is to deal with.

Using conditional independence

For each value $v = \text{R}, \text{G}, \text{B}$:

Condition on $X_1 = v$.

Find the maximum weight assignment (easy).



R 3

G 6

B 1

maximum weight is 6

- Now that we understand conditional independence, how is it useful?
- First, this formalizes the fact that if someone tells you the value of a variable, you can condition on that variable, thus potentially breaking down the problem into simpler pieces.
- If we are not told the value of a variable, we can simply try to condition on all possible values of that variable, and solve the remaining problem using any method. If conditioning breaks up the factor graph into small pieces, then solving the problem becomes easier.
- In this example, conditioning on $X_1 = v$ results in a fully disconnected graph, the maximum weight assignment for which can be computed in time linear in the number of variables.

Summary

Independence: when sets of variables A and B are disconnected; can solve separately.

Conditioning: assign variable to value, replaces binary factors with unary factors

Conditional independence: when C blocks paths between A and B

Markov blanket: what to condition on to make A conditionally independent of the rest.

- **Independence** is the key property that allows us to solve subproblems in parallel. It is worth noting that the savings is huge — exponential, not linear. Suppose the factor graph has two disconnected variables, each taking on m values. Then backtracking search would take m^2 time, whereas solving each subproblem separately would take $2m$ time.
- However, the factor graph isn't always disconnected (which would be uninteresting). In these cases, we can **condition** on particular values of a variable. Doing so potentially disconnects the factor graph into pieces, which can be again solved in parallel.
- Factor graphs are interesting because every variable can still influence every other variable, but finding the maximum weight assignment is efficient if there are small bottlenecks that we can condition on.