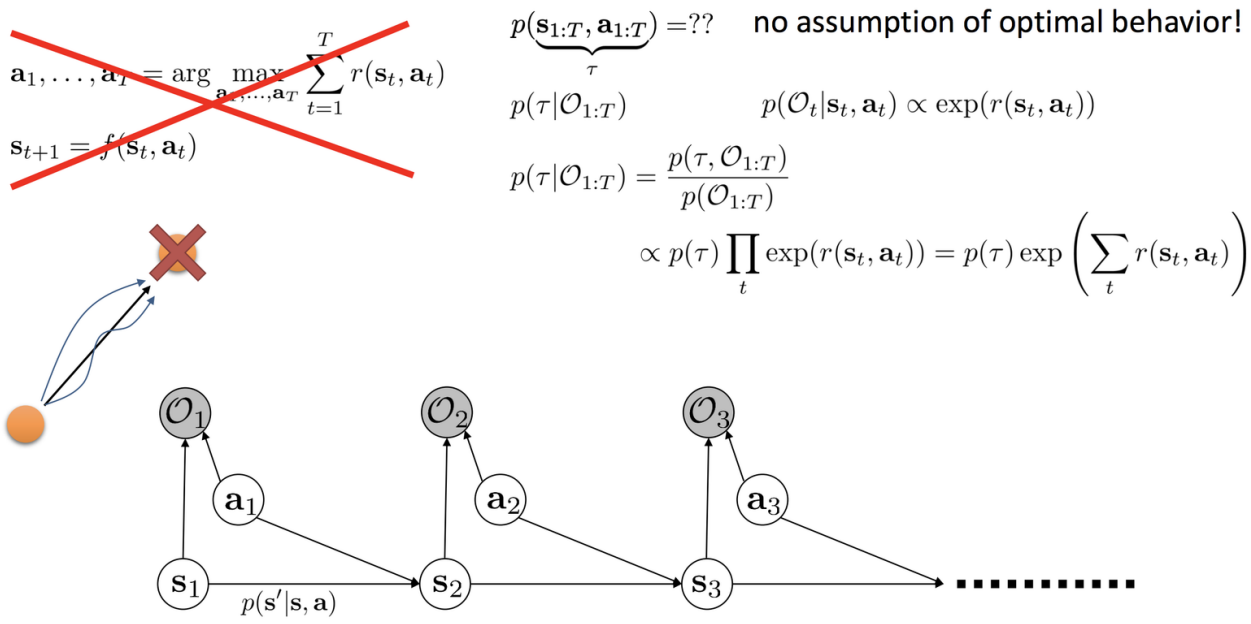
connection between control and inference

- are we humans going about tasks in an optimal manner? like do we use LQR or something? what is even our cost function? we are likely not being super optimal. maybe we are just approximately optimal and we are satisfied with that.
- instead of looking for crazy optimality can we use some stochastic approach to generate this kind of approximately optimal behavior? lets look at probabilistic graphical models (GMs) to generate approximately optimal behavior.
- we will use a model like the one below:



  note that each (state, action) can be optimal or not depending on the exponential of the reward *at that time step*. this means that our view of this optimality variable is limited to the current time step and not to the trajectory like we did in previous lectures.
- the advantage of the above model is that it explains sub-optimal/good enough behavior. this is because it makes any trajectory that is dynamically (physics-ally) possible and with a good total sum of rewards reasonably possible and ones with low ones exponentially less likely to occur.
- what is the advantage of the above formulation:
  - we can now explain suboptimal, but still good behavior
  - we have re-framed planning as an inference problem and we can use all our probabilistic graphical inference algorithms to solve these planning problems.
  - provides an explanation for stochastic behavior (this is different from the suboptimal property)
- the inference problem here is interested in answering three questions:

how to do inference?

1. compute backward messages $\beta_t(\mathbf{s}_t, \mathbf{a}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t)$
2. compute policy $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T})$
3. compute forward messages $\alpha_t(\mathbf{s}_t) = p(\mathbf{s}_t|\mathcal{O}_{1:t-1})$

- by doing some probability math involving conditional independence and marginalizing (nothing complicated just manipulation and reasoning that I won't go into here), Sergey derives how the backward message is basically like a Q-function (how likely are we to be optimal if we take a given action at a given state.
- the second (computing policy). Sergey uses the math derived in the previous step (that interprets this probability math as Q-function computation) to derive policies.
- the last, which is forward messages computes the probability of being in a state at time t given that we have been optimal from from time 1 to $t-1$. this is useful for inverse RL (maybe because he will assign rewards for states based on the probability of a state given that we have been optimal at all timesteps that got us here → $p(s_t|O_{1:t-1})$. Again, Sergey does some simple, but spaghetti math to derive the forward messages that I won't go into here.
- in summary, we have reframed the optimal control/reinforcement learning problem (a stochastic version of it) as probabilistic inference that we can use our probability toolkit to solve. as we do this, we get a "soft" version of q-learning where super sub-optimal trajectories have a very low likelihood (but not 0).
- The intuition is that this softness property can be good for initializations because they encourage more exploration than the hard version that tends to keep pursuing something kind of good it has found early on, thus making it more likely to drift towards a local optimum. If you see the term "entropy regularization" then this is the intuition behind it.