Branching Time Active Inference with Bayesian Filtering

Théophile Champion TMAC3@KENT.AC.UK

University of Kent, School of Computing

Canterbury CT2 7NZ, United Kingdom

Marek Grześ M.GRZES@KENT.AC.UK

University of Kent, School of Computing

Canterbury CT2 7NZ, United Kingdom

Howard Bowman H.BOWMAN@KENT.AC.UK

University of Birmingham, School of Psychology,

Birmingham B15 2TT, United Kingdom

University of Kent, School of Computing

Canterbury CT2 7NZ, United Kingdom

Editor: TO BE FILLED

Abstract

Branching Time Active Inference (Champion et al., 2022b,a) is a framework proposing to look at planning as a form of Bayesian model expansion. Its root can be found in Active Inference (Friston et al., 2016; Da Costa et al., 2020; Champion et al., 2021), a neuroscientific framework widely used for brain modelling, as well as in Monte Carlo Tree Search (Browne et al., 2012), a method broadly applied in the Reinforcement Learning literature. Up to now, the inference of the latent variables was carried out by taking advantage of the flexibility offered by Variational Message Passing (Winn and Bishop, 2005), an iterative process that can be understood as sending messages along the edges of a factor graph (Forney, 2001). In this paper, we harness the efficiency of an alternative method for inference called Bayesian Filtering (Fox et al., 2003), which does not require the iteration of the update equations until convergence of the Variational Free Energy. Instead, this scheme alternates between two phases: integration of evidence and prediction of future states. Both of those phases can be performed efficiently and this provides a forty times speed up over the state-of-the-art.

Keywords: Branching Time Active Inference, Bayesian Filtering, Free Energy Principle

1. Introduction

Active inference applies the free energy principle to generative models with actions (Friston et al., 2016; Da Costa et al., 2020; Champion et al., 2021) and can be regarded as a form of planning as inference (Botvinick and Toussaint, 2012). Over the years, this framework has successfully explained a wide range of phenomena, such as habit formation (Friston et al., 2016), Bayesian surprise (Itti and Baldi, 2009), curiosity (Schwartenbeck et al., 2018), and dopaminergic discharge (FitzGerald et al., 2015). It has also been applied to a variety of tasks such as navigation in the Animal AI environment (Fountas et al., 2020), robotic control (Pezzato et al., 2020; Sancaktar et al., 2020), the mountain car problem (Çatal et al., 2020), the game DOOM (Cullen et al., 2018) and the cart pole problem (Millidge, 2019).

However, because active inference defines the prior over policies as a marginal distribution over the space of all possible policies, the method suffers from an exponential space and time complexity class. In what follows, we will refer to this kind of active inference as “zero-tethered active inference" because all policies effectively start at time step zero. In the reinforcement learning literature, this exponential growth can be tackled using Monte Carlo tree search (MCTS) (Browne et al., 2012), whose origins can be found in the multi-armed bandit problem (Auer et al., 2002). More recently, MCTS has been applied to a large number of tasks such as the game of Go (Silver et al., 2016), the Animal AI environment (Fountas et al., 2020), and many others.

More recently, Branching Time Active Inference (BTAI) (Champion et al., 2022b,a) proposed that planning is a form of Bayesian model expansion guided by the upper confidence bound for trees (UCT) criterion from the MCTS literature, i.e. a quantity from the multi-armed bandit problem whose objective is to minimize the agent's regret. And because the generative model is dynamically expanded, variational message passing (VMP) (Winn and Bishop, 2005) was used to carry out inference over the latent variables. VMP can be understood as a flexible iterative process that sends messages along the edges of a factor graph (Forney, 2001), and computes posterior beliefs by summing those messages together.

Bayesian filtering (BF) (Fox et al., 2003) is an alternative inference method composed of two phases. In the first phase, Bayes theorem is used to compute posterior beliefs each time a new observation is obtained from the environment. In the second phase, posterior beliefs over the present state (S_t) are used to predict posterior beliefs over the state at the next time step (S_{t+1}). Note, that Bayesian filtering for discrete state space models is formally equivalent to belief propagation, i.e., both approaches lead to the same posterior beliefs. Importantly, Bayesian filtering is not iterative within a time step, i.e., it only contains a forward pass, and therefore is much more efficient than VMP, which contains both forward and backward messages.

In other words, we focus on the forward pass in state estimation; as opposed to the forward and backward passes found in Bayesian smoothing. This is the key distinction, from the perspective of the current work, because filtering is easier than smoothing. For example, in discrete state space models, forward-backward schemes usually rely upon variational message passing, as opposed to belief propagation. Having said this, it is possible to implement filtering or forward schemes using variational message passing for both discrete and continuous state space models. For an illustration of this in the context of generalised Bayesian filtering please see (Friston et al., 2017b).

In Section 2, we present the theory underlying Branching Time Active Inference when using Bayesian filtering for inference over latent variables. In Section 3, we show that using Bayesian filtering instead of variational message passing for the inference process provides BTAI with a forty times speed-up while maintaining effective planning. Finally, Section 5 concludes this paper and discusses avenues for future research.

2. Branching Time Active Inference with Bayesian Filtering ($\mathrm{BTAI_{BF}}$)

In this section, we describe the theory underlying our approach. For any notational uncertainty the reader is referred to Appendix F of Champion et al. (2022b). We let $\boldsymbol{D}$ be a 1-tensor representing the prior over initial hidden states $P(S_0)$. Let $\boldsymbol{A}$ be a 2-tensor representing the likelihood mapping $P(O_\tau|S_\tau)$, and $\boldsymbol{B}$ be a 3-tensor representing the transition mapping $P(S_{\tau+1}|S_\tau, U_\tau)$. Additionally, we let $\mathbb{I}$ be the set of multi-indices containing all the policies (i.e., sequences of actions) that have been explored by the model. The generative model of BTAI with BF can be formally written as the following joint distribution:

$$P(O_0, S_0, O_{\mathbb{I}}, S_{\mathbb{I}}) = P(O_0|S_0)P(S_0)\prod_{I \in \mathbb{I}} P(O_I|S_I)P(S_I|S_{I\setminus\text{last}})$$

where $S_{I\setminus\text{last}}$ is the parent of S_I, and:

$$\begin{aligned} P(S_0) &= \text{Cat}(\boldsymbol{D}) & P(O_\tau|S_\tau) &= \text{Cat}(\boldsymbol{A}) \\ P(O_I|S_I) &= \text{Cat}(\boldsymbol{A}) & P(S_I|S_{I\setminus\text{last}}) &= \text{Cat}(\boldsymbol{B}_I). \end{aligned}$$

where $\boldsymbol{B}_I = \boldsymbol{B}(\bullet, \bullet, I_{\text{last}})$ is the 2-tensor corresponding to I_{last} (i.e., the last action that led to S_I), and the likelihood mapping in the past, i.e., $P(O_\tau|S_\tau)$, and in the future, i.e., $P(O_I|S_I)$, are both categorical distributions with parameters $\boldsymbol{A}$. This generative model is depicted in Figure 1, where we assume that the current time step t equals zero.

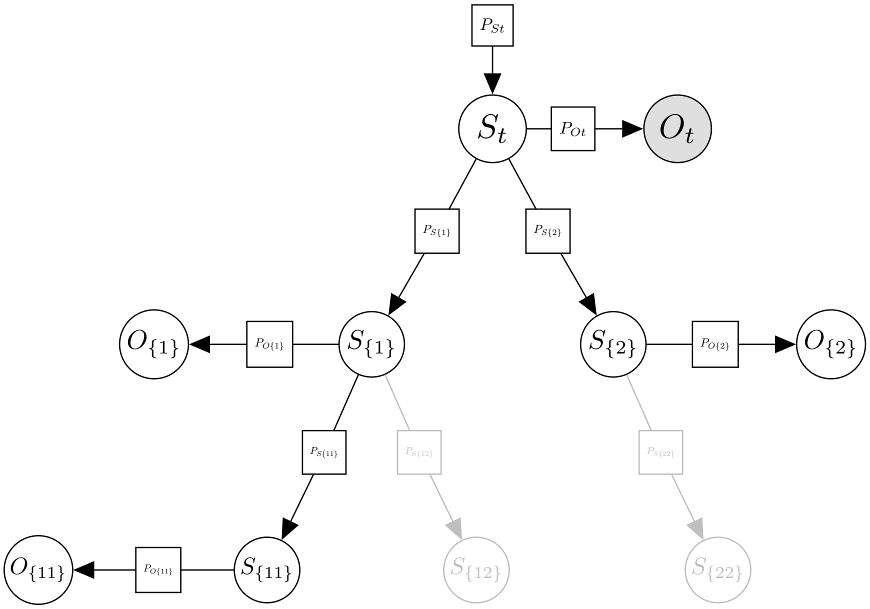

Figure 1: This figure illustrates the expandable generative model used by the BTAI with BF agent. The future is a tree like generative model whose branches correspond to the policies considered by the agent. The branches can be dynamically expanded during planning and the nodes in light gray represent possible expansions of the current generative model.

Initially, the generative model only contains the initial state S_0 and observation O_0. The prior over the hidden state is known, i.e. $P(S_0) = \text{Cat}(\boldsymbol{D})$, as well as the likelihood, i.e., $P(O_\tau|S_\tau) = \text{Cat}(\boldsymbol{A})$, and $P(O_0)$, the evidence, can be computed in the usual way by marginalizing over $P(O_0, S_0) = P(O_0|S_0)P(S_0)$. Thus, we can integrate the evidence provided to us by the initial observation O_0 using Bayes Theorem:

$$\mathcal{B}(S_0) = \frac{P(O_0|S_0)P(S_0)}{P(O_0)}, \tag{1}$$

where $\mathcal{B}(S_0)$ are the beliefs over the initial hidden state. Then, we use the UCT criterion to determine which node in the tree should be expanded. Let the tree's root

S_0 be called the current node. If the current node has no children, then it is selected for expansion. Alternatively, the child with the highest UCT criterion becomes the new current node and the process is iterated until we reach a leaf node (i.e. a node from which no action has previously been selected). The UCT criterion (Browne et al., 2012) for the j-th child of the current node is given by:

$$UCT_j = -\bar{\boldsymbol{G}}_j + C_{explore}\sqrt{\frac{\ln n}{n_j}}, \tag{2}$$

where $\bar{\boldsymbol{G}}_j$ is the average expected free energy calculated with respected to the actions selected from the j-th child, $C_{explore}$ is the exploration constant that modulates the amount of exploration at the tree level, n is the number of times the current node has been visited, and n_j is the number of times the j-th child has been visited. Importantly, the expected free energy (see below) is, effectively, the variational free energy expected under posterior predictive beliefs, under the action in question.

Let S_I be the (leaf) node selected by the above selection procedure. We then expand all the children of S_I, i.e., all the states of the form $S_{I::U}$ where $U \in \{1, ..., |U|\}$ is an arbitrary action, and $I :: U$ is the multi-index obtained by appending the action U at the end of the sequence defined by I. Next, we compute the predicted beliefs over those expanded hidden states using the transition mapping:

$$\mathcal{B}(S_J) = \mathbb{E}_{\mathcal{B}(S_I)}\big[P(S_J|S_I)\big], \tag{3}$$

where we let $J = I :: U$ for any action U, $\mathcal{B}(S_I)$ are the predicted posterior beliefs over S_I, and according to our generative model $P(S_J|S_I) = \text{Cat}(\boldsymbol{B}_J)$ with $\boldsymbol{B}_J = \boldsymbol{B}(\bullet, \bullet, J_{\text{last}})$. The above equation corresponds to the second phase of Bayesian filtering, i.e., the prediction phase, which involves the calculation of new beliefs, using the generative model, in the absence of new observations. Then, we need to estimate the cost of (virtually) taking each possible action. The cost in this paper is taken to be the expected free energy (Friston et al., 2017a):

$$\boldsymbol{G}_J \triangleq D_{\mathrm{KL}}[\mathcal{B}(O_J)||V(O_J)] \; + \; \mathbb{E}_{\mathcal{B}(S_J)}[\mathrm{H}[P(O_J|S_J)]], \tag{4}$$

where the prior preferences over future observations are specified by the modeller as $V(O_J) = \mathrm{Cat}(\boldsymbol{C})$, according to the generative model $P(O_J|S_J) = \mathrm{Cat}(\boldsymbol{A})$, and the posterior beliefs over future observations are computed by prediction as follows:

$$\mathcal{B}(O_J) = \mathbb{E}_{\mathcal{B}(S_J)}[P(O_J|S_J)].$$

Next, we assume that the agent will always perform the action with the lowest cost, and back-propagate the cost of the best (virtual) action toward the root of the tree. Formally, we write the update as follows:

$$\forall K \in \mathbb{A}_I \cup \{I\}, \quad \boldsymbol{G}_K \leftarrow \boldsymbol{G}_K + \min_{U \in \{1,\ldots,|U|\}} \boldsymbol{G}_{I::U}, \tag{5}$$

where I is the multi-index of the node that was selected for (virtual) expansion, and $\mathbb{A}_I$ is the set of all multi-indices corresponding to ancestors of S_I. During the back propagation, we also update the number of visits as follows:

$$\forall K \in \mathbb{A}_I \cup \{I\}, \quad n_K \leftarrow n_K + 1. \tag{6}$$

If we let $\boldsymbol{G}_K^{aggr}$ be the aggregated cost of an arbitrary node S_K obtained by applying Equation 5 after each expansion, then we are now able to express $\bar{\boldsymbol{G}}_K$ formally as:

$$\bar{\boldsymbol{G}}_K = \frac{\boldsymbol{G}_K^{aggr}}{n_K}.$$

The planning procedure described above ends when the maximum number of planning iterations is reached, and the action corresponding to the root's child with the lowest average cost is performed in the environment. At this point, the agent receives a new observation O_τ and needs to update its beliefs over S_τ. First, we predict the posterior beliefs over S_τ as follows:

$$\mathcal{B}(S_\tau | U_{\tau-1} = U^*) = \mathbb{E}_{\mathcal{B}(S_{\tau-1})}\big[P(S_\tau | S_{\tau-1}, U_{\tau-1} = U^*)\big], \tag{7}$$

where U^* is the action performed (from the root) in the environment, $P(S_\tau | S_{\tau-1}, U_{\tau-1} = U^*)$ is the 2-tensor $\boldsymbol{B}(\bullet, \bullet, U^*)$, and $\mathcal{B}(S_{\tau-1})$ is the agent's posterior beliefs over the state at time $\tau - 1$, e.g., after performing the first action in the environment, $\tau = 1$ and $\mathcal{B}(S_{\tau-1}) = \mathcal{B}(S_0)$ as given by Equation 1. Second, we integrate the evidence provided by the new observation O_τ using Bayes theorem:

$$\mathcal{B}(S_\tau) = \frac{P(O_\tau | S_\tau)\mathcal{B}(S_\tau | U_{\tau-1} = U^*)}{P(O_\tau)}, \tag{8}$$

where $\mathcal{B}(S_\tau | U_{\tau-1} = U^*)$ is used as an empirical prior. By an empirical prior we mean a posterior distribution of the previous time step, e.g., $\mathcal{B}(S_\tau | U_{\tau-1} = U^*)$, that is used as a prior in Bayes theorem. Algorithm 1 concludes this section by summarizing our approach.

Algorithm 1: BTAI with BF: action-perception cycles (with relevant equations indicated in round brackets).

```
Input: $env$ the environment, $O_0$ the initial observation, $\boldsymbol{A}$ the likelihood
       mapping, $\boldsymbol{B}$ the transition mapping, $\boldsymbol{C}$ the prior preferences, $\boldsymbol{D}$ the
       prior over initial states, $N$ the number of planning iterations, $M$ the
       number of action-perception cycles.
$\mathcal{B}(S_0) \leftarrow$ IntegrateEvidence($O_0$, $\boldsymbol{A}$, $\boldsymbol{D}$)                    // Using (1)
$root \leftarrow$ CreateTreeNode(beliefs = $\mathcal{B}(S_0)$, action = -1, cost = 0, visits = 1)
 // Where -1 in the line above is a dummy value
repeat $M$ times
    repeat $N$ times
        $node \leftarrow$ SelectNode($root$)                    // Using (2) recursively
        $eNodes \leftarrow$ ExpandChildren($node$, $\boldsymbol{B}$)        // Using (3) for each
         action
        Evaluate($eNodes$, $\boldsymbol{A}$, $\boldsymbol{C}$)        // Compute (4) for each expanded
         node
        Backpropagate($eNodes$)                              // Using (5) and (6)
    end
    $U^* \leftarrow$ SelectAction($root$)     // Such that $U^*$ minimises the average
     cost
    $O_\tau \leftarrow env$.Execute($U^*$)
    $\mathcal{B}(S_{\tau-1}) \leftarrow root.beliefs$                // Get beliefs of the root node
    $\mathcal{B}(S_\tau | U_{\tau-1} = U^*) \leftarrow$ ComputeEmpiricalPrior($\boldsymbol{B}$, $\mathcal{B}(S_{\tau-1})$, $U^*$)   // Using
     (7)
    $\mathcal{B}(S_\tau) \leftarrow$ IntegrateEvidence($O_\tau$, $\boldsymbol{A}$, $\mathcal{B}(S_\tau | U_{\tau-1} = U^*)$)        // Using (8)
    $root \leftarrow$ CreateTreeNode(beliefs = $\mathcal{B}(S_\tau)$, action = $U^*$, cost = 0, visits =
     1)
end
```

3. Results

In this section, we first present the deep reward environment in which two versions of BTAI will be compared. Then, we present experimental results comparing BTAI with VMP and BTAI with BF in terms of running time and performance.

3.1 Deep reward environment

This environment is called the deep reward environment because the agent needs to navigate a tree-like graph where the graph's nodes correspond to the states of the system, and the agent needs to look deep into the future to diferentiate the good path from the traps. At the beginning of each trial, the agent is placed at the root of the tree, i.e., the initial state of the system. From the initial state, the agent can perform $n + m$ actions, where n and m are the number of good and bad paths, respectively. Additionally, at any point in time, the agent can make two observations: a pleasant one or an unpleasant one. The states of the good paths produce pleasant observations, while the states of the bad paths produce unpleasant ones.

If the first action selected was one of the m bad actions, then the agent will enter a bad path in which $n + m$ actions are available at each time step but all of them produce unpleasant observations. If the first action selected was one of the n good actions, then the agent will enter the associated good path. We let L_k be the length of the k-th good path. Once the agent is engaged on the k-th path, there are still $n + m$ actions available but only one of them keeps the agent on the good path. All the other actions will produce unpleasant observations, i.e., the agent will enter a bad path.

This process will continue until the agent reaches the end of the k-th path, which is determined by the path's length L_k. If the k-th path was the longest of the n good paths, then the agent will from now on only receive pleasant observations independently of the action performed. If the k-th path was not the longest path, then independently of the action performed the agent will enter a bad path.

To summarize, at the beginning of each trial, the agent is prompted with n good paths and m bad paths. Only the longest good path will be beneficial in the long term, the others are traps, which will ultimately lead the agent to a bad state. Figure 2 illustrates this environment.

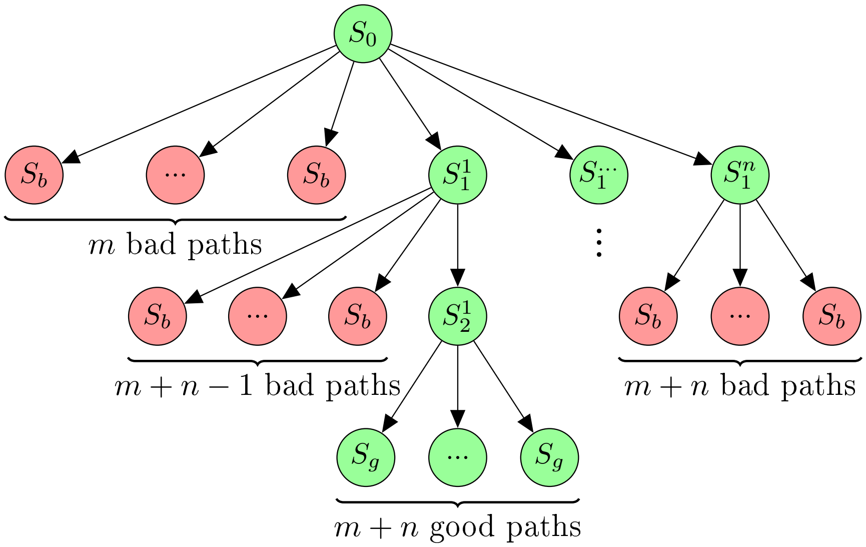

Figure 2: This figure illustrates a type of deep reward environment where S_0 represents the initial state, S_b represents a bad state, S_g represents a good state, and S_j^i is the j-th state of the i-th good path. Also, the longest path in the above picture is the first good path whose length L_1 is equal to two. Importantly, the longest path corresponds to the only good path that does not turn out to be a trap.

3.2 BTAI with VMP versus BTAI with BF

In this section, we compare BTAI with VMP and BTAI with BF in terms of running time and performance. The running time reported in Tables 1 and 2 was obtained by running 100 trials each composed of 20 action-perception cycles. Also, the trial was stopped whenever the agent reached a bad state or the goal state. As shown in Tables 1 and 2, both approaches were able to solve the tasks. However, BTAI with BF ran around forty times faster than BTAI with VMP.

This speed up is possible for two reasons. First, Bayesian filtering does not require the iteration of the belief updates until convergence of the variational free energy. Second, when computing the optimal posterior over a random variable X, VMP needs to compute one message for each adjacent variable of X, add them together, and normalise using a softmax function. In contrast, BF only performs a forward pass, which is essentially implemented as matrix multiplications.

n	m	L_1, L_2, ..., L_n	# planning iterations	P(goal)	P(bad)	Running time (ms)
2	5	5, 8	25	1	0	23.42 ± 4.966
2	5	5, 8	50	1	0	43.37 ± 2.743
2	5	5, 8	100	1	0	110.58 ± 29.586
3	5	6, 5, 8	25	1	0	26.7 ± 2.405
3	5	6, 5, 8	50	1	0	61.41 ± 45.112
3	5	6, 5, 8	100	1	0	120.25 ± 12.722

Table 1: This table presents the results of BTAI with BF on various deep reward environments. Recall, that n and m are the number of good and bad paths, respectively. L_i is the length of the i-th good path. $P(goal)$ reports the probability of reaching the goal state (i.e., the agent successfully picked the longest path), and $P(bad)$ reports the probability of reaching the bad state (i.e., either by picking a bad action directly of by falling into a trap).

n	m	L_1, L_2, ..., L_n	# planning iterations	P(goal)	P(bad)	Running time (ms)
2	5	5, 8	25	1	0	924.63 ± 38.930
2	5	5, 8	50	1	0	1908.54 ± 60.621
2	5	5, 8	100	1	0	4574.45 ± 301.197
3	5	6, 5, 8	25	1	0	1038.43 ± 51.322
3	5	6, 5, 8	50	1	0	2119.19 ± 19.866
3	5	6, 5, 8	100	1	0	5342.04 ± 252.953

Table 2: This table presents the results of BTAI with VMP on various deep reward environments. Recall, that n and m are the number of good and bad paths, respectively. L_i is the length of the i-th good path. $P(goal)$ reports the probability of reaching the goal state (i.e., the agent successfully picked the longest path), and $P(bad)$ reports the probability of reaching the bad state (i.e., either by picking a bad action directly of by falling into a trap).

4. Discussion

In this section, we discuss how our work relates to others in the literature. We have discussed various computational architectures for active (planning as) inference — and have shown that the current scheme, based upon Bayesian filtering, is the most efficient. To place the current scheme in relation to other active inference schemes, it is helpful to think about the distinctions in terms of the underlying generative model. Active inference can be regarded as the Bayesian inversion of a generative model that includes the consequences of action. Operationally, this means that active inference can be defined as selecting action sequences (i.e., policies) that minimise the variational free energy (or maximise marginal likelihood) expected, when committing to a policy.

However, this does not specify the particular form of the generative model. There are two sorts of generative models used in active inference. The first, considers a policy that starts at the beginning of a trial (Friston et al., 2017b), i.e., zero-tethered active inference. The second uses a generative model in which policies start at the current time, as described in the present paper. The requisite belief updating for these two kinds of models is fundamentally different. In the first kind, evidence that the agent is pursuing a particular policy accumulates during the execution of that policy. In the second kind, every policy is, a priori, equally plausible at the point of evaluation.

In active inference of the first kind, information during policy execution informs the likelihood that this policy is currently being enacted. In turn, this requires belief updating about past states, to assess the likelihood of observations in the past. This mandates backward message passing from the present to the past and an implicit form of working memory. In contrast, active inference of the second kind can proceed using belief updating into the future.

In the terminology of state estimation, this means zero-tethered active inference involves forward and backward algorithms (e.g., variational message passing), while active inference of the second kind just requires forward message passing or belief propagation. Variational message passing and belief propagation are procedures found in generative models of discrete states. The equivalent belief updating in generative models of continuous states is generally referred to as Bayesian smoothing (with forward and backward passes) and Bayesian filtering (with just forward passes). In short, because our deep tree search starts from the present, it only needs the filtering or forward pass. Technically, this is important because one can replace variational message passing (required for smoothing or forward and backward passes) with belief propagation in the forwards direction, which is much more efficient.

Belief propagation refers to propagating posterior beliefs about the present into the future, using suitable probability transition matrices. We have leveraged this simplification and efficiency, much in the spirit of the deep tree searches described in terms of sophisticated inference (Friston et al., 2021). Although numerical experiments may be needed to confirm this picture, it suggests that the improvement in computational efficiency — demonstrated in the current work — rests upon finessing the inference problem through a commitment to belief propagation — as opposed to the more demanding problem of representing the past and implicit abilities for postdiction.

5. Conclusion and future works

In this paper, we proposed a new implementation of Branching Time Active Inference (Champion et al., 2022b,a), where the inference is carried out using Bayesian filtering (Fox et al., 2003), instead of using variational message passing (Champion et al., 2021; Winn and Bishop, 2005).

This new approach has a few advantages. First, it achieves the same performance as its predecessor around forty times faster. Second, the implementation is simpler and less data structures need to be stored in memory.

Also, one could argue that there is a trade-off in the nature and extent of the information inferred by zero-tethered active inference, branching-time active inference with variational message passing (BTAI_{VMP}) from Champion et al. (2022b,a), and branching-time active inference with Bayesian Filtering (BTAI_{BF}). Specifically, zero-tethered active inference exhaustively represents and updates all possible policies, while BTAI_{VMP} will typically only represent one policly in the past (i.e., the one undertaken by the agent) and a small subset of the possible (future) trajectories. These will typically be the more advantageous paths for the agent to pursue, with the less beneficial paths not represented at all. Indeed, the tree search is based on the expected free energy that favors policies that maximize information gain, while realizing the prior preferences of the agent. BTAI_{BF} stores even less data than BTAI_{VMP}, because the sequence of past hidden states is discarded as time passes, and only the beliefs over the current and future states are stored.

Additionally, full variational inference can update the system's understanding of past contingencies on the basis of new observations. As a result, the system can obtain more refined information about previous decisions, perhaps re-evaluating the optimality of these past decisions. Because zero-tethered active inference represents a larger space of policies, this re-evaluation could apply to more policies. When using Bayesian filtering, beliefs about past hidden states are discarded as time progresses, which makes Bayesian belief updating (about past hidden states) impossible.

We also know that humans engage in counterfactual reasoning (Rafetseder et al., 2013), which, in our planning context, could involve the entertainment and evaluation of alternative (non-selected) sequences of decisions. It may be that, because of the more exhaustive representation of possible trajectories, zero-tethered active inference can more efficiently engage in counterfactual reasoning. In contrast, branching-time active inference would require these alternative pasts to be generated "a fresh" for each counterfactual deliberation. In this sense, one might argue that there is a trade-off: branching-time active inference provides considerably more efficient planning to attain current goals, zero-tethered active inference provides a more exhaustive assessment of paths not taken. In contrast, branching time active inference implemented with Bayesian filtering does not leave a memory at all, let alone one upon which conterfactual reasoning could be realized.

The implementation of Branching Time Active Inference with variational message passing can be found here: `https://github.com/ChampiB/Homing-Pigeon`, and the implementation of Branching Time Active Inference with Bayesian Filtering is available on Github: https://github.com/ChampiB/Branching_Time_Active_Inference.

Even with this forty times speed up, BTAI is still unable to deal with large scale observations such as images. Adding deep neural networks to approximate the likelihood mapping is therefore a compelling direction for future research.

Also, this framework is currently limited to discrete action and state spaces. Designing a continuous extension of BTAI would enable its application to a wider range of problems such as robotic control with continuous actions. Note that creating generative models of continuous processes is probably best achieved by equipping a discrete state space model (of the sort used above), with a level that maps to continuous state spaces. In brief, this involves specifying a continuous trajectory as a succession of fixed points that are generated by a Markov decision process. See Friston et al. (2017b), for an example. This means that one could apply BTAI, in principle, to real-world, continuous state space problems, such as robotics and active vision.

Finally, as the depth of the tree increases, the beliefs about future states tend to become more and more uncertain, which can lead to a drop in performance. This suggests that there exists an optimal number of planning iterations, after which the model simply does not have enough information to keep planning. Future work could thus focus on automatically identifying this optimal number of planning iterations, in order to improve the robustness of the approach.

Acknowledgments

We would like to thank the reviewers for their valuable feedback, which greatly improved the quality of the paper.

References

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256.

Botvinick, M. and Toussaint, M. (2012). Planning as inference. *Trends in Cognitive Sciences*, 16(10):485 – 488.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. (2012). A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43.

Champion, T., Bowman, H., and Grześ, M. (2022a). Branching time active inference: Empirical study and complexity class analysis. *Neural Networks*.

Champion, T., Da Costa, L., Bowman, H., and Grześ, M. (2022b). Branching time active inference: The theory and its generality. *Neural Networks*, 151:295–316.

Champion, T., Grześ, M., and Bowman, H. (2021). Realizing Active Inference in Variational Message Passing: The Outcome-Blind Certainty Seeker. *Neural Computation*, 33(10):2762–2826.

Cullen, M., Davey, B., Friston, K. J., and Moran, R. J. (2018). Active inference in OpenAI Gym: A paradigm for computational investigations into psychiatric illness. *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, 3(9):809 – 818. Computational Methods and Modeling in Psychiatry.

Da Costa, L., Parr, T., Sajid, N., Veselic, S., Neacsu, V., and Friston, K. (2020). Active inference on discrete state-spaces: A synthesis. *Journal of Mathematical Psychology*, 99:102447.

FitzGerald, T. H. B., Dolan, R. J., and Friston, K. (2015). Dopamine, reward learning, and active inference. *Frontiers in Computational Neuroscience*, 9:136.

Forney, G. D. (2001). Codes on graphs: normal realizations. *IEEE Transactions on Information Theory*, 47(2):520–548.

Fountas, Z., Sajid, N., Mediano, P., and Friston, K. (2020). Deep active inference agents using monte-carlo methods. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 11662–11675. Curran Associates, Inc.

Fox, V., Hightower, J., Liao, L., Schulz, D., and Borriello, G. (2003). Bayesian filtering for location estimation. *IEEE Pervasive Computing*, 2(3):24–33.

Friston, K., Da Costa, L., Hafner, D., Hesp, C., and Parr, T. (2021). Sophisticated Inference. *Neural Computation*, 33(3):713–763.

Friston, K., FitzGerald, T., Rigoli, F., Schwartenbeck, P., Doherty, J. O., and Pezzulo, G. (2016). Active inference and learning. *Neuroscience & Biobehavioral Reviews*, 68:862 – 879.

Friston, K., FitzGerald, T., Rigoli, F., Schwartenbeck, P., and Pezzulo, G. (2017a). Active Inference: A Process Theory. *Neural Computation*, 29(1):1–49.

Friston, K. J., Parr, T., and de Vries, B. (2017b). The graphical brain: Belief propagation and active inference. *Network Neuroscience*, 1(4):381–414.

Itti, L. and Baldi, P. (2009). Bayesian surprise attracts human attention. *Vision Research*, 49(10):1295 – 1306. Visual Attention: Psychophysics, electrophysiology and neuroimaging.

Millidge, B. (2019). Combining active inference and hierarchical predictive coding: A tutorial introduction and case study. *PsyArXiv*.

Pezzato, C., Hernandez, C., and Wisse, M. (2020). Active inference and behavior trees for reactive action planning and execution in robotics. *arXiv*.

Rafetseder, E., Schwitalla, M., and Perner, J. (2013). Counterfactual reasoning: From childhood to adulthood. *Journal of experimental child psychology*, 114(3):389–404.

Sancaktar, C., van Gerven, M., and Lanillos, P. (2020). End-to-end pixel-based deep active inference for body perception and action. *arXiv*.

Schwartenbeck, P., Passecker, J., Hauser, T. U., FitzGerald, T. H. B., Kronbichler, M., and Friston, K. (2018). Computational mechanisms of curiosity and goal-directed exploration. *bioRxiv*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Winn, J. and Bishop, C. (2005). Variational message passing. *Journal of Machine Learning Research*, 6:661–694.

Çatal, O., Verbelen, T., Nauta, J., Boom, C. D., and Dhoedt, B. (2020). Learning perception and planning with deep active inference. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3952–3956.