Towards Explainable MCTS

Hendrik Baier, Michael Kaisers
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
hendrik.baier@cwi.nl, michael.kaisers@cwi.nl

Abstract

Monte-Carlo Tree Search (MCTS) is a family of sampling-based search algorithms widely used for online planning in sequential decision-making domains, and at the heart of many recent breakthroughs in AI. Understanding the behavior of MCTS agents is non-trivial for developers and users, as it results from often large and complex search trees, consisting of many simulated possible futures, their evaluations, and relationships to each other. This paper is presenting our ongoing exploration of possible explanations for MCTS decision-making and behavior. It is for the first time trying to tackle some of the challenges previously posed for explainable search, which include: meaningfully summarizing the space of possible futures spanned by the available actions of the AI and their possible consequences, in order to explain the AI's choices between them; considering such explanations not only as static objects but as interactive conversations between user and AI; and understanding explanation not only as a one-way information flow from the AI to the user, but as a tool for human-AI collaboration and for leveraging both AI and human capabilities in problem solving.

Introduction

The field of Explainable AI has so far mainly been focused on explaining data-driven systems, such as neural networks trained through supervised learning (Guidotti et al. 2019; Henin and Métayer 2019). Recently there have been increasing efforts on explaining goal-driven systems as well, such as agents acting in complex, sequential decision-making problems (Sado et al. 2020). But even in such settings, often suitable for the framework of reinforcement learning (RL), most prior work on explanations for RL is concerned with reactive, neural-network based agents (Lam et al. 2020).

Going beyond the capabilities of such reactive agents, some of the most successful RL approaches at the heart of recent AI breakthroughs have been powered by *online planning* or *search* (Silver et al. 2017b; 2017a; Schrittwieser et al. 2019), often using variants of *Monte Carlo Tree Search (MCTS)* (Kocsis and Szepesvári 2006). Sampling-based search algorithms such as MCTS tackle huge search spaces through selective search, provably converge to optimal behavior in many settings, provide approximate solutions when stopped at any time, and recently also proved to show excellent synergy with integrated deep neural networks (Silver et al. 2017b; Anthony, Tian, and Barber 2017). They are now being developed for and applied to countless domains from board and video games to robotics to drug discovery to self-driving cars and more (Claes et al. 2017; Chan et al. 2019; Hoel et al. 2020). However, the decision-making and behavior of MCTS can be hard to understand, as it results from often large and complex search trees, consisting of thousands of simulated possible futures, their evaluations, and relationships to each other. This can be a challenge e.g. for developers trying to debug and improve MCTS; for end users who need to build trust in and successfully collaborate with MCTS-based agents to deploy solutions in practice; and for the development of completely new applications, such as turning algorithms that can play chess on a superhuman level into helpful and effective chess teachers.

The contribution of this paper is to share our progress towards explainable MCTS. Our bottom-up approach, consisting of concrete building blocks of MCTS explanations that can be expanded and adapted for a variety of explanation scenarios, complements recent work taking a top-down approach in outlining the open challenges of explainable search (Baier and Kaisers 2020). We make first steps towards tackling several of these challenges. First, since decisions can be considered intelligent to the extent that their expected consequences achieve our objectives (Russell 2019), for our explanations we draw on the search tree over expected futures available to MCTS. Our toolset aims at explaining current decisions by explicitly discussing possible future situations and further decisions that given choices could lead to. Second, due to the complexity of search trees and the multitude of questions a human user could have about the future plans of MCTS, our toolset aims at understanding explanations as interactive conversations between user and AI agent, rather than giving one-size-fits-all explanations. And third, due to the nature of online planning, in which usually neither the AI nor the user can consistently make optimal decisions on their own, our toolset suggests simple ways to "discuss" algorithm decisions before committing to them – aiming at explanations that support true human-AI collaboration.

While the proposed elements of explainable MCTS can be adapted to very different domains, throughout this article we describe examples for two board game domains: Connect Four and Breakthrough[1]. The application scenario could for example be a beginner player trying to learn the game through explanations of observed, stronger MCTS play; a developer trying to find flaws in the reasoning of an MCTS agent by requesting explanations; or even a player using explanations to more effectively play in a team with the MCTS agent, as a simple model for collaborative tasks involving humans and AI.

Elements of explainable MCTS

In this section, we first discuss the framing and goals of our work in order to situate it in the related literature on explainable AI and planning, before presenting concrete elements of MCTS explanations that compose the proposed toolset.

When attempting to explain search algorithms – understanding them essentially as "black boxes" that take a sequential decision-making problem as input, and return an action decision as output – one can choose different basic approaches that are to some degree similar to choices in explaining black box machine learning models. One could for example try to build search algorithms that are more explainable *by design*: algorithms that are restricted to small enough search trees, and use a simple enough understanding of their search domains, that we can consider them inherently interpretable, e.g. in the sense of simulatable (Lipton 2018), by humans. An example could be a chess algorithm that thinks only two moves ahead, and uses only piece counts for the evaluation of positions. In our work however, we stick to using the original search algorithms in their full complexity and decision-making strength, and focus instead on providing explanations for the reasons behind particular algorithmic decisions by means of external XAI techniques[2].

An interesting and challenging aspect of explaining MCTS is that it cannot be cleanly divided along the lines of algorithm-focused, model-focused, and plan-focused explanations as suggested by previous research into explainable AI planning (Chakraborti, Sreedharan, and Kambhampati 2020). When acting online, typically without being able to achieve optimality in limited time, model-focused explanations for example cannot fully explain why a decision was made while staying entirely agnostic of the search algorithm that generated it. Even different runs of the same MCTS algorithm on the same problem might construct different trees and come up with different decisions.

Instead, all of our MCTS explanations are based on the search tree, which

1. contains the current expectations and plans for the future from which the algorithm's action decision is derived, allowing for partly plan-focused explanations. Plan explanation techniques such as summarization and abstraction are used by all elements of MCTS explanations presented here – applied to the multitude of branching plans represented by the tree, not just a single plan as typically assumed in the explainable planning literature.

2. contains the relevant understanding that MCTS has of the model/domain, such as which future actions are legal, which future states can result etc., allowing for partly model-based explanations. We do not yet attempt to explicitly formalize the mental model or the inferential capability of the user here, but MCTS explanations address an assumed difference in inferential capability between AI and user any time they attempt to simplify the tree. By walking the explainee through important possible futures that were considered by MCTS, some common model differences such as states that were misevaluated, or legal actions that were forgotten by the explainee, are addressed as well.

3. contains information on which possible futures were explored to which depth by MCTS, which possible actions were sampled how often, and the resulting statistics used for internal processing, allowing for partly algorithm-focused explanations. However, we do not walk the explainee step by step through MCTS search decisions like a debugger, but aim at higher-level explanations such as: "I thought a lot about possible consequences of action X, but not as much about action Y because it seemed less promising to me. My uncertainty is therefore higher here."

As we do not formalize the user model or the user's inferential capabilities in this preliminary work, we also do not commit yet to a definition of "optimal", such as for example "minimally sufficient" (Khan, Poupart, and Black 2009), explanations. In fact, dealing with unknown user models and capabilities is even more challenging for MCTS than in the AI planning setting (Sreedharan, Srivastava, and Kambhampati 2018): Due to the countless possible (imperfect) models that could (indeterministically) result in any given (suboptimal) action decision in a sequential decision-making problem, a user for example asking "Why should I make move *d3* here instead of *b3*?" does not give us much information on their thinking process to work with. We leave such user modelling, potentially involving extended human-AI interactions, as future work.

Instead, drawing from our experience on how to explain moves in a board game, we suggest that explaining MCTS decisions boils down to two fundamental processes. Given a search tree $T = (N, E)$ consisting of nodes N (often representing future states) and edges E (often representing future actions), a typical explanation performs

1. *tree simplification* – reducing the number of possible futures presented to the user, by picking the most relevant subsets $N_r \subset N$ and $E_r \subset E$ to integrate into an explanation. In order to explain why move *d3* is preferred to move *b4*, for example, the simplest contrastive explanation could include only the root node, the *fact* move *d3* preferred by the search algorithm, and the *foil* move *b4* suggested by the user, as well as the states resulting from both. Depending on

[1]For the rules of Breakthrough, see (Lorentz and Horey 2013).

[2]This has also been called "post-hoc explainability" in the literature (Arrieta et al. 2020), but we would like to avoid confusion with what we call post-hoc explanations in this paper: explanations given after a search process is completed.

the complexity of the choice between *b4* and *d3*, more future states and move choices could be included. Note that also for other tree structures such as decision trees, a reduction in tree size has been found to contribute crucially to interpretability (Arrieta et al. 2020); and that the choice of most relevant future states and actions has similarities to the choice of *interestingness elements* as proposed for explaining entire RL policies (Sequeira and Gervasio 2019).

2. *subtree summarization* – reducing the complexity of the information presented to the user about N_r and E_r for the explanation at hand. This is essentially achieved by summarizing or abstracting the subtrees under the elements of N_r and E_r, as they represent everything the algorithm knows about the relevant states and actions and their future expected consequences. For example, the states following move *d3* could have an interpretable feature such as the safety of a certain piece on the board, which the states following *b4* do not have; or the opposing player could be likely to achieve a subgoal of the game as a (potentially delayed) consequence of *b4*, but is unlikely to do so after *d3*. In both cases, a good explanation provides arguments for *d3* and against *b4* by summarizing positive expected outcomes of *d3* and negative expected outcomes of *b4* for the user.

In this paper, the information we present on any given subtree is defined by the probabilities of a pre-specified, domain-dependent set of positive and negative *scenarios* (sets of states), as defined by the proportion of MCTS simulations in the subtree that reached them. The scenarios are assumed to be shared knowledge among AI and human user. The feature of piece safety mentioned earlier for example defines a positive scenario for the user, and the opponent achieving a known subgoal of the game defines a negative scenario. Using sets of preferred states, chosen such that "users understand that this is a good thing" (Khan, Poupart, and Black 2008), is a relatively common method in XAI to ground explanations; it has been argued that users need to be given explanations using domain-like language (Cruz, Dazeley, and Vamplew 2019), as opposed to the expected action values and future rewards used internally by many RL algorithms. This is also a matter of the right level of abstraction: When using higher-level concepts such as scenarios, "these concepts can be designed to be more descriptive and informative for the potential user" (van der Waa et al. 2018), because "humans naturally interpret a plan as achieving abstract tasks (or subgoals)" (Zhang et al. 2017). We leave the automatic learning of such scenarios, for example by extracting them from neural networks, for future work.

Note that we could make use of different modalities for presenting explanations; text, voice, visualizations, and others. How to effectively choose and integrate different modalities is one of the current challenges for explainable goal-driven agents (Sado et al. 2020), with promising results already being demonstrated in explainable ML (Park et al. 2018). In explainable search, different modalities can be used both for explanations that refer to the search tree – which nodes and edges were expanded, how often they were sampled or evaluated, etc. – as well as for explanations that are projected on the application domain – in a board game for example, explanations in terms of the board and the pieces and moves on it. In this work, we are primarily focusing on text when explaining elements of the tree, and visualizations such as heatmaps when explaining aspects of the domain, but this is still an open research question.

In the following, we propose various approaches to MCTS explanation, divided into two categories: techniques that explain MCTS decisions after the search has finished, and collaborative explanation techniques that allow for the user to actively participate in the decision-making process.

Post hoc explanations (after the search)

These are explanation techniques where the query does not influence the decision of the search algorithm. In a first phase, MCTS searches and decides on a best course of action – in a board game for example, on the next move *b2*; and in an independent second phase, it then explains its decision and its reasoning leading to that decision – answering the question, "Why *b2*?" Post hoc explanations can be one-off explanations as in most of the existing XAI literature, or they can be structured as conversations, giving the user opportunity to drill into specific points of interest ("Why not *c3* instead of *b2*? What happens if the opponents answers *b2* with *d4*?"). Depending on the application scenario, the user can then use their new insights into the decision-making of MCTS to judge the quality of the decision, learn from MCTS, debug MCTS, decide how much to trust MCTS, decide whether to go with the decision of MCTS or not, etc.; but the search process itself remains independent of this.

We found three different basic types of post hoc explanations to arise naturally across many explanation settings. They correspond to the following user queries (**bold**) or commands (`monospaced`), which we have currently implemented for our two example board game domains:

"Why do you recommend this action?" After MCTS has returned an action, the command `explain decision` simply asks for an explanation of why this action was found to be best. Implicitly, this can be seen as a request for a contrastive explanation, comparing the algorithm's action choice to all alternative legal actions in the current state. An explanation of the recommended action can also subsume more general information on the context of the decision: in a game for example, how the game is progressing, what the most likely next positions and moves are, what the currently expected final outcome of the game is, etc. – all of which can support the action decision. This is the most basic request for explanation from a search algorithm, and does not yet involve any back-and-forth interaction between user and AI, although it can be a useful entry point into one.

Our current implementations of `explain decision` give their explanations in three parts[3]. In the first part, they present the moves considered most likely in the next couple of timesteps, by simplifying the tree to the principal variation (PV) and its sibling nodes. Parameters determine how deep into the future the PV is presented, and how many siblings (alternatives) are shown per move. Expected futures

[3]For a full example explanation from a Connect Four game, see https://bit.ly/3eJ25fK

are shown both in term of sampling statistics from the tree (algorithm-focused), as well as in terms of what they look like on the board (domain-focused). Along with the expected future boards, text templates are combined with statistics extracted from the tree in order to describe current expectations for the game outcome (e.g. "I'm pretty sure we are winning here"), as well as to compare legal moves, based on confidence intervals of their expected values (e.g. "The best choice for the opponent seems to be e2, although a2 could very well be equal. d3, f3, b5, c2 are definitely worse; and g3 is probably worse."). In order to explore a possible interpretability-completeness tradeoff (Gilpin et al. 2018), we also implemented a higher-detail variant in which alternative sibling moves are not only mentioned, but their resulting boards are also shown and labeled with summaries of their consequences (subtree summarization) compared to the PV move. In Connect Four for example, we show which winning groups of 4 discs would become more or less likely for the user and their opponent if a given alternative move was taken, by averaging over all possible futures MCTS simulated through the respective moves.

In the second part, the context of the decision is given by applying a domain-dependent form of subtree summarization to the root node. In Connect Four, that includes listing the overall most probable winning groups for both players (shown on the board); and the correlation of each board square with winning, to show where the players should focus their attention (as a heatmap). In Breakthrough, it includes the correlation of each piece surviving with the players winning the overall game, as extracted from all MCTS simulations; and the expected material balance between the players over the next moves. These features serve to make the user aware of salient aspects of the current game situation.

In the third part, the recommended move is explicitly contrasted with the union of all alternative moves, using summarization of the subtrees below these moves at the root. In Connect Four, it is shown on which squares both players are most likely to win, and how those probabilities differ between simulated games going through the recommended move vs. any other move. The same comparison is shown for the probabilities of owning each square at game end (or within the horizon simulated by MCTS). It is also shown how the recommended move changes the most likely winning groups for both players. In Breakthrough, we show comparisons between the recommended move and all other moves in terms of the probabilities of each piece getting captured; of each piece winning the game; of each goal square getting reached; and on how far both players are expected to get on the board. All of these are simple concepts for Breakthrough players, which allow for some insight on the influence of one move on the overall development of the game.

"Why don't you recommend this alternative action?" After MCTS has returned an action, the command `why not` <alternativeAction> is a request for a more focused, explicit contrastive explanation: Why does the search algorithm consider its recommended action (the *fact* action) to be better than the specific alternative action given by the user as an argument to the command (the *foil* action)? Because the foil is only a single action here, the possible consequences of executing it can typically be explained and compared to the fact action in more detail than when looking at *all* alternative actions as e.g. in the overview given by the `explain decision` command. The tree is simplified to fewer nodes, which allows for more detail in the summarization of the subtrees under those nodes.

Our current implementations of `why not` first show the most likely future moves and their most-sampled alternatives (siblings) both for the fact move and for the foil move – the PV, and the PV if the first move was replaced by the foil. After comparing their consequences in terms of expected specific future boards and moves, the moves are then compared in terms of aggregate future expectations (subtree summarization): We show comparisons for the same game-specific scenarios as outlined for the third part of the `explain decision` command, only now comparing one move to another instead of one move to all others.

"What do you recommend in these possible futures?" With the command `explore`, the user enters an interactive "exploration mode", allowing the entire search tree built by the finished search to be studied. The exploration mode starts at the root, and then lets the user traverse any desired branch and request corresponding explanations. In contrast to the previous two commands, which exclusively focus on the decision in the root node by explaining and contrasting its most likely consequences, this interactive mode allows for such questions as, "What if this specific sequence of actions happened next? Have you studied that case, how would you judge that situation, and what would you do?" The exploration mode also allows the user to get a better idea of the limits of the search algorithm, as the tree can be explored all the way down to the leaves, where only a few simulations have formed typically highly uncertain expectations. The commands currently provided for the exploration mode are `if` <action>, which shifts the focus of the algorithm down to the child node corresponding to the given action; `back` which shifts it back to the parent node of the current node; `root` which returns all the way to the starting point at the root (to reset the exploration); and `quit` which leaves the exploration node. At any time, the user can use the previously outlined `explain decision` and/or `why not` <move> commands to better understand the particular future situation and decision in the currently explored node.

This exploration mode is a first step towards "exploring [the space of possible futures] together with the user", which has been described as one of the unique challenges of explainable search (Baier and Kaisers 2020). Making natural use of the tree structure built by MCTS, it builds on the insight that "in real-world problems where the system may be planning for multiple eventualities in stochastic environments (...) it needs to be able to present the policy at a high-level of abstraction and delve into details as required" (Sreedharan, Srivastava, and Kambhampati 2020). It also has similarities to one of the first attempts at iterative dialogue for explainable AI planning, which traverses trees of contrastive explanations (Cashmore et al. 2019), as opposed to the underlying data structure.

Collaborative explanations (influencing the search)

These are explanations where information can flow both ways between user and AI. There are no distinct search and explanation phases here; instead, the user has the opportunity to interact with MCTS before the algorithm's final decision is made, for example by pointing out potentially underexplored actions that seem promising to the user, or by requesting to re-do the search under specific constraints that the user expects to hold true. Collaborative explanations do not only allow the user to learn about and from the algorithm, but also to influence it in return – which can lead to synergetic performance in domains where human-AI teamwork is crucial (Akata et al. 2020). The search algorithm might for example be more suited to quickly considering large numbers of possible futures and aggregating their probabilities, while the human collaborator might be aware of how to avoid situations in which the AI is known to struggle or even blunder, complementing the AI's abilities.

For our two example board game domains, we have currently implemented three different initial approaches to such collaborative explanations. They correspond to the following user requests:

"If you thought more about this alternative action, would you change your recommendation and why/why not?" The command `think about` <alternativeAction> can be used after MCTS has returned an action, requesting additional search with a changed focus. It is motivated by the fact that MCTS algorithms often search quite selectively. This can occasionally lead MCTS to ignore one action in favor of another for a long time during search, simply because the latter got "luckier" in the first few times it was sampled. The `think about` command forces MCTS to spend some additional search effort on a particular alternative action at the root. It can be useful in cases where the user believes that the alternative action might have been "unlucky" so far, and deeper search of it might lead to MCTS ultimately preferring it to the "lucky" currently preferred action. The answer to the `think about` command is an explanation of what the altered focus changes or does not change about the algorithm's decision. In our current implementations, the PVs of the original and the extended search are presented; it is explained with the help of a text template whether the additional search has changed the preferred move or not; and a contrastive explanation of the original preferred move and the suggested move is given, with similar content to a `why not` comparison.

In future work, this approach could be extended to the user recommending actions for additional exploration anywhere in the search tree, not only at the root ("Why don't you think more about the opponent's option of answering *b3* with *a4*? I think that could change our current plan").

"If you expected this opponent behavior, would you change your recommendation and why/why not?" The command `expect response` <action2> `to` <action1> is similar to `think about` <alternativeAction> in that it also requests additional search effort from the MCTS algorithm. However, instead of recommending specific possible futures for additional exploration, the `expect response` command instead specifies constraints under which the entire search should be re-done: In our particular implementation, the constraint that the user's action1 would always be followed by the opponent's action2. This can be useful in two-player games when the human user realizes that action2 is the obvious refutation of action1 (think of an immediate recapture in chess that invalidates the idea of capturing in the first place). The search can be sped up considerably if MCTS does not have to discover that refutation itself. More generally, the `expect response` command allows the user to align certain expectations with the search algorithm by conveying a simple prior belief – here on expected opponent behavior. After receiving this command and conducting the modified search, MCTS will then produce an explanation, contrasting the unmodified and modified search results. We implemented this explanation equivalent to a `think about` explanation, except that the foil move is not a parameter of the user command anymore, but the preferred move returned by the modified search.

In our current implementation, such expected responses have to be specified for concrete game states, without the ability to generalize ("Try always recapturing when this knight is captured, regardless of the current game state"). Constraints are also currently always hard constraints, not soft constraints that can be used as guidance for the search without being hard and fast rules. We leave these extensions, as well as alternative forms of constraints or suggestions, as future work; these could include constraints on the agent's own behavior, or expectations on the behavior of a stochastic environment to resolve any model mismatch.

"If I helped you search, would you change your recommendation and why/why not?" The command `search together` initiates a "hybrid problem-solving" mode, in which the search for a best action in the current state is repeated – but MCTS is now allowed to ask the user for their input while searching. In our implementation, the search proceeds as normal, until a possible future state is recognized as important for the current decision-making process. A simple heuristic for importance, for example, is how often a future state has been visited by MCTS so far, i.e. how likely it is to happen according to MCTS. When reaching a given visit threshold at the state, MCTS can then interrupt the search; present an explanation of the future state and the problem it poses together with a preliminary analysis of how the algorithm would expect to act in it; and directly ask the user if they have a strong intuition themselves for what should be done. If the user does have such an intuition, it will from now on be used as a constraint in the search, analogously to the constraints given by the `expect response` command. If the user is not sure what would happen in that possible future, the search simply proceeds without additional constraint. When the search is finished, the results are then presented in the form of a contrastive explanation, juxtaposing the results of the initial AI-only search and the results of the later human-AI collaborative search analogously to a `expect response` explanation.

A variant of `search together` has been implemented that only requests information from the user when the relevant future decision is not obvious enough for the algorithm itself, to minimize effort on the side of the user. A variant that uses human inputs not as hard constraints, but as guidance that can still be overruled if the search later discovers it to be suboptimal, is remaining as future work.

In this way, the decision-making process of MCTS can be boosted by actively requesting human knowledge where it might be of value, and integrating it into the search where available. This simple approach is a first step towards a joint search for the best policy that makes optimal use of both the algorithm's and the user's capabilities. In the future, these capabilities could be formalized to integrate them even more effectively into a *hybrid intelligence* (Akata et al. 2020).

Related Work

There is little prior research on using search trees for behavior explanation. For improving trust and team performance in a collaborative human-robot task where the robot uses a "bounded lookahead procedure" in every timestep, various templates for natural-language explanations have been explored (Wang, Pynadath, and Hill 2016). This work can be seen as a first step into the direction we propose, but was dealing with 1-step lookahead only, as opposed to the explanations for large search trees in complex domains envisioned here. It therefore did not have to make use of any tree simplification or summarization techniques.

During the preparation of this paper, two further publications approached the topic of explaining online tree search. In order to identify flaws in a planning-based deep RL agent playing a strategy game, an interesting tree visualization has been proposed that directly integrates a simplified view of the game state into the displayed corresponding tree nodes (Lam et al. 2020). Similar to our "exploration mode", the user is starting out from the principal variation, and is then given the opportunity to interactively explore the rest of the tree. However, a strongly pruned 2-step lookahead was used in this work, which simplified the tree presentation and made subtree summarization unnecessary.

For the goal of allowing human users to learn simulated curling strategies from a tree search algorithm, visual justifications for algorithm decisions have been proposed consisting of various action outcomes that are visually most similar to the expected action outcome, but most different in terms of expected end-game value (Silva, Lelis, and Bowling 2020). This is meant to teach the user how specific state features affect expected outcomes. While the search algorithm used in this work is able to search deeper than 1 timestep, all action outcomes presented to the user are only 1 step in the future, and no information is given about additional decisions or possible futures they could lead to.

Our research also has partially overlapping goals with prior work on visualizations of heuristic search (Magnaguagno et al. 2017), which does not consider our online setting, but provides some information such as the overall shape of the search tree that was needed to solve a problem to optimality, and the heuristic evaluations of its states. In contrast, our work is providing a first conceptual overview of explanation types and techniques for MCTS.

Future Work

In this paper, we presented a variety of tools that we expect to be of use in ongoing and future work on explaining the decisions and behavior of MCTS-like algorithms. A wide range of directions for such future work remains.

In term of capabilities, it would be interesting to expand the scope of MCTS explanations from single to multiple decisions or entire episodes of behavior, such as complete games in our test domains. Such longer interactions with users might also involve changing interests, needs, and expectations on the side of the user, and therefore bring the social aspect of explanations (Miller 2019), user-awareness and personalization, stronger into focus. In addition, several open challenges of explainable search have not yet been tackled by the tools proposed here – these include for example counterfactual explanations of why MCTS did *not* explore certain branches more deeply, as well as explanations for where MCTS "changed its mind" during the search process, by discovering that initial estimates had been overly optimistic or pessimistic (Baier and Kaisers 2020). Furthermore, the research area of *mixed-initiative planning* could inspire explanations that support deeper models of collaboration, in which users and AIs do not only engage in conversations that are structured and led by either the user or the AI, but also allow the initiative to shift back and forth between them, and allow their contributions to be negotiated more freely as a task is being solved (Cohen et al. 1998; Hearst 1999; Jiang and Arkin 2015).

In term of evaluation, work in two orthogonal directions will be crucial for further progress on explainable MCTS and explainable search: First, a robust and flexible formalization of explainable search will be useful in order to structure research efforts better, and derive potentially important concepts for *sufficient* and *optimal* explanations (compare e.g. (Khan, Poupart, and Black 2009)). This could take a similar form to the recently evolving formalization of explainable classical planning (Chakraborti et al. 2019), and might allow for explicit model and inference reconciliation. Second, carefully constructed user studies (Doshi-Velez and Kim 2017; Buçinca et al. 2020) will be important in order to learn more about what are preferred and effective explanations of search in practice, and about how to satisfy actual user needs with a library of tools such as developed in our ongoing work. In our game domains for example, user testing could help evaluate which MCTS explanations in fact best help users improve their collaboration with AI teammates, increase their game-playing skill (Silva, Lelis, and Bowling 2020), or help them understand different search-based agents with their respective strengths and weaknesses (Lam et al. 2020), to name just a few possible scenarios.

Acknowledgments

We would like to thank the reviewers for their valuable input. This work is part of the project *Flexible Assets Bid Across Markets* (FABAM, project number TEUE117015), funded within the Dutch Topsector Energie / TKI Urban Energy by Rijksdienst voor Ondernemend Nederland (RvO).

References

Akata, Z.; Balliet, D.; de Rijke, M.; Dignum, F.; Dignum, V.; Eiben, G.; Fokkens, A.; Grossi, D.; Hindriks, K.; Hoos, H.; Hung, H.; Jonker, C.; Monz, C.; Neerincx, M.; Oliehoek, F. A.; Prakken, H.; Schlobach, S.; van der Gaag, L. C.; van Harmelen, F.; van Hoof, H.; van Riemsdijk, B.; van Wynsberghe, A.; Verbrugge, R.; Verheij, B.; Vossen, P.; and Welling, M. 2020. A Research Agenda for Hybrid Intelligence: Augmenting Human Intellect With Collaborative, Adaptive, Responsible, and Explainable Artificial Intelligence. *Computer* 53:18–28.

Anthony, T.; Tian, Z.; and Barber, D. 2017. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems*, 5360–5370.

Arrieta, A. B.; Rodríguez, N. D.; Ser, J. D.; Bennetot, A.; Tabik, S.; Barbado, A.; García, S.; Gil-Lopez, S.; Molina, D.; Benjamins, R.; Chatila, R.; and Herrera, F. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion* 58:82–115.

Baier, H., and Kaisers, M. 2020. Explainable Search. In *2020 IJCAI-PRICAI Workshop on Explainable Artificial Intelligence*.

Buçinca, Z.; Lin, P.; Gajos, K. Z.; and Glassman, E. L. 2020. Proxy tasks and subjective measures can be misleading in evaluating explainable AI systems. In Paternò, F.; Oliver, N.; Conati, C.; Spano, L. D.; and Tintarev, N., eds., *IUI '20: 25th International Conference on Intelligent User Interfaces, Cagliari, Italy, March 17-20, 2020*, 454–464. ACM.

Cashmore, M.; Collins, A.; Krarup, B.; Krivic, S.; Magazzeni, D.; and Smith, D. E. 2019. Towards Explainable AI Planning as a Service. *CoRR* abs/1908.05059.

Chakraborti, T.; Kulkarni, A.; Sreedharan, S.; Smith, D. E.; and Kambhampati, S. 2019. Explicability? Legibility? Predictability? Transparency? Privacy? Security? The Emerging Landscape of Interpretable Agent Behavior. In *29th International Conference on Automated Planning and Scheduling (ICAPS 2018)*, 86–96.

Chakraborti, T.; Sreedharan, S.; and Kambhampati, S. 2020. The Emerging Landscape of Explainable AI Planning & Decision Making. *CoRR* abs/2002.11697.

Chan, H. C. S.; Shan, H.; Dahoun, T.; Vogel, H.; and Yuan, S. 2019. Advancing Drug Discovery via Artificial Intelligence. *Trends in Pharmacological Sciences* 40(8):592–604.

Claes, D.; Oliehoek, F. A.; Baier, H.; and Tuyls, K. 2017. Decentralised Online Planning for Multi-Robot Warehouse Commissioning. In Larson, K.; Winikoff, M.; Das, S.; and Durfee, E. H., eds., *16th Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2017)*, 492–500. ACM.

Cohen, R.; Allaby, C.; Cumbaa, C.; Fitzgerald, M.; Ho, K.; Hui, B.; Latulipe, C.; Lu, F.; Moussa, N.; Pooley, D.; Qian, A.; and Siddiqi, S. 1998. What is Initiative? *User Model. User Adapt. Interact.* 8(3-4):171–214.

Cruz, F.; Dazeley, R.; and Vamplew, P. 2019. Memory-Based Explainable Reinforcement Learning. In *32nd Australasian Joint Conference on Advances in Artificial Intelligence*, volume 11919 of *Lecture Notes in Computer Science*, 66–77.

Doshi-Velez, F., and Kim, B. 2017. Towards A Rigorous Science of Interpretable Machine Learning. *CoRR* abs/1702.08608.

Gilpin, L. H.; Bau, D.; Yuan, B. Z.; Bajwa, A.; Specter, M.; and Kagal, L. 2018. Explaining Explanations: An Overview of Interpretability of Machine Learning. In *5th IEEE International Conference on Data Science and Advanced Analytics (DSAA 2018)*, 80–89.

Guidotti, R.; Monreale, A.; Ruggieri, S.; Turini, F.; Giannotti, F.; and Pedreschi, D. 2019. A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv.* 51(5):93:1–93:42.

Hearst, M. A. 1999. Trends Controversies: Mixed-initiative interaction. *IEEE Intell. Syst.* 14(5):14–23.

Henin, C., and Métayer, D. L. 2019. Towards a generic framework for black-box explanations of algorithmic decision systems. In *IJCAI 2019 Workshop on Explainable Artificial Intelligence*.

Hoel, C.; Driggs-Campbell, K. R.; Wolff, K.; Laine, L.; and Kochenderfer, M. J. 2020. Combining Planning and Deep Reinforcement Learning in Tactical Decision Making for Autonomous Driving. *IEEE Trans. Intell. Veh.* 5(2):294–305.

Jiang, S., and Arkin, R. C. 2015. Mixed-Initiative Human-Robot Interaction: Definition, Taxonomy, and Survey. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, 954–961. IEEE.

Khan, O. Z.; Poupart, P.; and Black, J. P. 2008. Explaining recommendations generated by MDPs. In *2008 ECAI Workshop on Explanation-aware Computing*, 13–24.

Khan, O. Z.; Poupart, P.; and Black, J. P. 2009. Minimal Sufficient Explanations for Factored Markov Decision Processes. In *19th International Conference on Automated Planning and Scheduling (ICAPS 2009)*.

Kocsis, L., and Szepesvári, C. 2006. Bandit Based Monte-Carlo Planning. In *17th European Conference on Machine Learning (ECML 2006)*, volume 4212 of *Lecture Notes in Computer Science*, 282–293.

Lam, K.-H.; Lin, Z.; Irvine, J.; Dodge, J.; Shureih, Z. T.; Khanna, R.; Kahng, M.; and Fern, A. 2020. Identifying Reasoning Flaws in Planning-Based RL Using Tree Explanations. In *2020 IJCAI-PRICAI Workshop on Explainable Artificial Intelligence*.

Lipton, Z. C. 2018. The mythos of model interpretability. *Commun. ACM* 61(10):36–43.

Lorentz, R., and Horey, T. 2013. Programming Breakthrough. In *8th International Conference on Computers and Games (CG 2013)*, volume 8427 of *Lecture Notes in Computer Science*, 49–59.

Magnaguagno, M. C.; Pereira, R. F.; Móre, M. D.; and Meneguzzi, F. 2017. WEB PLANNER: A tool to develop classical planning domains and visualize heuristic state-space search. In *2017 Workshop on User Interfaces and Scheduling and Planning (UISP 2017)*, 32–38.

Miller, T. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.* 267:1–38.

Park, D. H.; Hendricks, L. A.; Akata, Z.; Rohrbach, A.; Schiele, B.; Darrell, T.; and Rohrbach, M. 2018. Multimodal Explanations: Justifying Decisions and Pointing to the Evidence. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, 8779–8788. IEEE Computer Society.

Russell, S. 2019. *Human Compatible: Artificial Intelligence and the Problem of Control*. New York: Viking.

Sado, F.; Loo, C. K.; Kerzel, M.; and Wermter, S. 2020. Explainable Goal-Driven Agents and Robots - A Comprehensive Review and New Framework. *CoRR* abs/2004.09705.

Schrittwieser, J.; Antonoglou, I.; Hubert, T.; Simonyan, K.; Sifre, L.; Schmitt, S.; Guez, A.; Lockhart, E.; Hassabis, D.; Graepel, T.; Lillicrap, T. P.; and Silver, D. 2019. Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. *CoRR* abs/1911.08265.

Sequeira, P., and Gervasio, M. T. 2019. Interestingness Elements for Explainable Reinforcement Learning: Understanding Agents' Capabilities and Limitations. *CoRR* abs/1912.09007.

Silva, C. R.; Lelis, L. H. S.; and Bowling, M. 2020. Teaching Humans with Justifications of Monte Carlo Tree Search Decisions. In *2020 IJCAI-PRICAI Workshop on Explainable Artificial Intelligence*.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; Lillicrap, T. P.; Simonyan, K.; and Hassabis, D. 2017a. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *CoRR* abs/1712.01815.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; Chen, Y.; Lillicrap, T.; Hui, F.; Sifre, L.; van den Driessche, G.; Graepel, T.; and Hassabis, D. 2017b. Mastering the game of Go without human knowledge. *Nature* 550:354–359.

Sreedharan, S.; Srivastava, S.; and Kambhampati, S. 2018. Hierarchical Expertise Level Modeling for User Specific Contrastive Explanations. In *27th International Joint Conference on Artificial Intelligence (IJCAI 2018)*, 4829–4836.

Sreedharan, S.; Srivastava, S.; and Kambhampati, S. 2020. TLdR: Policy Summarization for Factored SSP Problems Using Temporal Abstractions. In *2020 International Conference on Automated Planning and Scheduling (ICAPS 2020)*.

van der Waa, J.; van Diggelen, J.; van den Bosch, K.; and Neerincx, M. A. 2018. Contrastive Explanations for Reinforcement Learning in terms of Expected Consequences. *CoRR* abs/1807.08706.

Wang, N.; Pynadath, D. V.; and Hill, S. G. 2016. The Impact of POMDP-Generated Explanations on Trust and Performance in Human-Robot Teams. In *2016 International Conference on Autonomous Agents & Multiagent Systems*, 997–1005.

Zhang, Y.; Sreedharan, S.; Kulkarni, A.; Chakraborti, T.; Zhuo, H. H.; and Kambhampati, S. 2017. Plan Explicability and Predictability for Robot Task Planning. In *2017 IEEE International Conference on Robotics and Automation (ICRA 2017)*, 1313–1320.