Planning in the midst of chaos: how a stochastic Blood Bowl model can help to identify key planning features

Jérémie Humeau, Alexis Lebis, Mathieu Vermeulen and Guillaume Lozenguez
IMT Lille Douai, Institut Mines-Télécom, Univ. Lille, Centre for Digital Systems
F-59000 Lille, France
e-mail: firstname.lastname@imt-lille-douai.fr

***Abstract*—For several decades now, games have become an important research ground for artificial intelligence. In addition to often present useful and complex problems, they also provide a clear framework thanks to their rules, sometimes numerous. In this article, we explore a very difficult two-players board game named *Blood Bowl*. This game allows the players to perform many different actions, which depend for a large part on the result of one or more dice rolls. Thus, it can be seen as a multi-action probabilistic problem driven by a Markov decision process. In this article, we present the first stochastic model of the main phase of Blood Bowl to our knowledge and the premise of a dedicated planning framework. Such a framework could offer interesting grounds and insights for modeling high turn-wise branch factor games.**

***Index Terms*—Game study, Blood Bowl, Markov Decision Problem, model, planning, AI**

I. Introduction

Recently, *Blood Bowl* (Games Workshop, 1986) was introduced as a new board game challenge for AI [1]. Like *Chess* or *Go*, *Blood Bowl* is a strategy turn-based board game where two players fight each other. As these games, *Blood Bowl* is also a perfect information game. However, the significant differences in *Blood Bowl* come from the number of actions each player can perform at each turn and the introduction of probabilistic outcomes. Those features make this game very difficult for artificial intelligence, even with a reduced version of it (e.g. predetermined composition of each team, smaller boards).

For comparison, the average estimated *turn-wise* branch factor for *Chess* is 30 and 300 for *Go* [2], while for *Blood Bowl* it is estimated to 10^{50} [1]. Traditional model-based planning algorithm (*e.g.* Monte-Carlo tree search) and machine learning algorithms do not perform well for this type of large state-space probabilistic problems [3]. Thus, it is important to give these algorithms a framework to work with.

In this paper, we present a model of the *Blood Bowl*'s rules for simulating state progression, which does not exist yet to our knowledge. More specifically, we propose to model only the main phase of the game from a player's actions standpoint. The Fig. 1 shows the phase of the game we are focused on, compared to the course of an entire *Blood Bowl* game. This main phase is the playing phase: the two opponents, each in turns, perform various actions through players; their goal is to score more touchdowns than the other opponent. The other phases are highly specific to the game, and some, like Team Building, are complex *per se* and should be studied separately. Therefore, it is not the goal of this paper to propose any *Blood Bowl* solving algorithm.

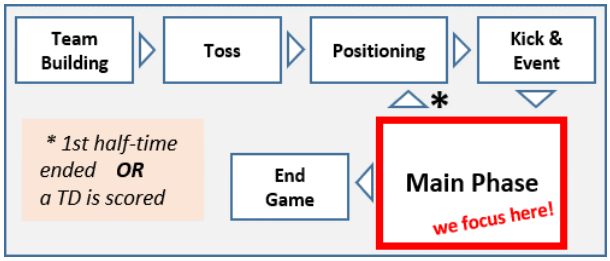

Fig. 1. An entire course of a *Blood Bowl* game. The main phase is where two opponents – named coaches – compete against each other by performing various players' actions to be the one who scores the most points *via* touchdowns (TD).

Our two main motivations are: 1) scoring situations are scarce in the state space of *Blood Bowl*. As shown in [1], 0 point was scored in 350,000 random games, suggesting that randomized search approaches would not lead to challenging AI for *Blood Bowl* [4], [5]. Thus, it is important to model and manage both accurate rules definition and expert knowledge for planning ; 2) *Blood Bowl*'s rules are complex: consequently it is necessary to establish a comprehensive and modeled baseline to facilitate the involvement of the community.

In the following section II, we introduce some game notions, required for the understanding of the paper. Then, we propose in section III a *Blood Bowl* main phase model. Afterwards, we present in section IV how such a model can infuse planning considerations through a multi layered framework. Finally, we conclude the paper in section V.

II. Glossary

Throughout this paper, we will consider the complete rules defined in the Living Rulebook 6 (LRB6) [6] because of its use as a reference for the majority of *Blood Bowl* competitions.

978-1-6654-3886-5/21/$31.00 ©2021 IEEE

LRB6 is about sixty pages long: we cannot review the rules here; a game overview and the main rules are listed in [1]. We will just make a few reminders all along this article.

Below are important notions and terminologies for the rest of this paper:

Game board: It is a grid of 15×26 squares.

Roster: A roster is the race of a team. There is twenty-four different official rosters. Each one has its specific players.

Team: *Blood Bowl* is a match between two teams. A team has at least eleven players, each having a kind according to its roster. In this paper, 1) team A has a yellow color and team B a blue color, and 2) team A is always initiating the action.

Coach: A coach is the human or AI who takes decisions during the game and controls the players.

Player: An entity controllable by a coach. A player has some basic skills induced by its roster, and up to six additional skills depending on the game style and rules.

Skill: A special action that can be triggered by a player during a match. There are 75 different skills regrouped in 6 categories: General (14), Agility (10), Passing (7), Strength (10), Mutation(10) and Extraordinary (24). Extraordinary skills are often associated with specific player kind (*e.g.* Big Guys).

Attribute: A player has four attributes: Movement (Mv), Agility (Ag), Strength (S) and Armor (Ar). A skill outcome may depend on the attributes of the player who use it.

Touchdown (TD): A touchdown is analogous to a try in rugby or a US football touchdown. In *Blood Bowl*, a team that puts a TD scores one point. At the end of a match, the team with the highest score wins. It is possible to have a draw.

Block dice (BD): A block dice is a particular 6 sided dice in Blood Bowl, as shown in Fig. 2. It is used when a player blocks an opponent player.

xDy: It is a dice roll with x the number of dice and y the type of dice used. Examples: 2D6 means 2 dice with 6 sides are rolled. 1D8 means 1 die with 8 sides is rolled.

$(-)x$BD: It is a roll of x block dice, "$-$" is used when the choice is done by the opposing coach.

x+: x is the minimum expected outcome of a dice roll for being successful. Example: 3+ needs 3 or more on a dice (dice roll for those results are often 1D6 or 2D6 and for 2D6 we take the sum of the 2 dice).

Reroll (RR): A reroll gives the possibility to redo a dice roll (a.k.a.re-roll). Note that a roll dice can never be re-rolled more than one time.

Turnover: A turnover is a critical aspect of *Blood Bowl*. Whenever an action fails, it finishes the coach turn.

Go For It (GFI): It is the possibility to move of 2 extra squares. As a general rule, for each extra square a 2+ result on 1D6 is needed.

III. Model Design

According to its rules, and that it is a perfect information game, *Blood Bowl* can be seen as a probabilistic planning problem where the two coaches have to plan actions for each of their players [7]. Indeed, several outcomes of the players' actions made by the coaches are stochastic (depending on dice rolls). Therefore, we propose to model this problem by using a Markov Decision Processes (*MDP*) [8].

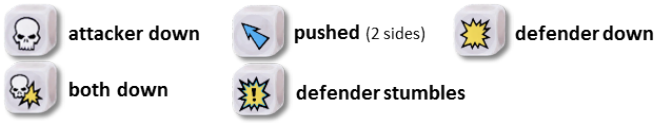

Fig. 2. Block dice (BD) sides.

A *MDP* is a quadruplet $\langle S, A, T, R \rangle$, with S a set of states, A a set of actions, T is a transition function $T : S \times A \times S \rightarrow [0; 1]$, where $T(s, a, s')$ represents the probability to be in state s', by doing action a while being previously in a state s. R is a reward function such that $R : S \times A \times S \rightarrow \mathbb{R}$ returning the reward (or the cost if negative) linked to the transition. Solving a *MDP* consists in defining policy $\pi : S \rightarrow A$ associating an action to perform to each state. The policy is optimal if it permits to maximize the expected gains (*e.g.* the sum of the rewards weighted by the probability to collect them in the future). Computing an optimal policy becomes intractable when the number of states increases [4], [5], [9].

In *Blood Bowl*, a state $s \in S$ corresponds to a "snapshot" of the game at any moment (*e.g.* position of the players, of the ball, team scores). The actions A represent all the actions available during a game that a coach can perform, conditioned by the game rules. One of the particularities of *Blood Bowl* is that a coach can perform several actions successively, for up to all its players, at each turn. The transition function T defines the effect of the action taken by a coach on the game's state. As mentioned previously, several of those actions (including the more critical ones) are conditioned by a dice roll.

Finally, R defines a reward (positive or negative) received by a coach. As a general definition for now, we define the reward function R only for the final states of the game. This reward function returns -1, 1 or 0, respectively if the coach loses the game, wins it, or in case of a draw.

Proposing specific reward functions is, in our opinion, too premature and does a disservice to the establishment of a theoretical framework. Indeed, considering that it would be difficult to plan on a distant horizon, resulting policies will probably tend to favor rapid scoring approaches – with potentially dramatic decisions for the rest of the game (*e.g.* the death of a player) – even if only a one point difference is enough to win the game.

The rest of the modeling is further described below.

A. State consideration and state modeling

As a reminder, we only focus on the modeling of the main phase and its states S, that is where a coach plans its players' actions. In this phase, a state s can be represented by all the elements on the game board. These elements can be separated into two sets: permanent elements P and variable elements V.

The permanent elements set P is composed of players' attributes and players' skills (cf. section II), cheerleaders, assistant coaches, babes and fame.

The variable elements set V is composed of the following elements: the position of all the players (a square on the pitch,

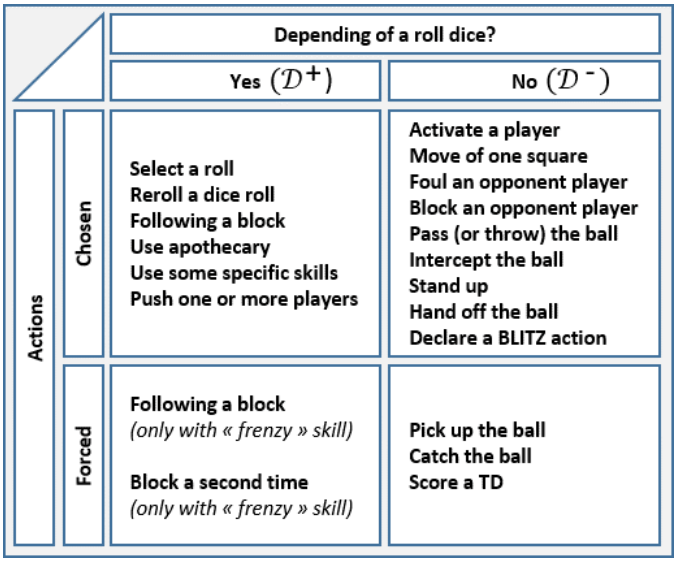

Fig. 3. Classification of primitive actions.

in the reserve or in the injury ground), status of the players on the pitch (stunned, knock down, standing up), current player availability (played, available, or currently playing), unique RR status from the current player's skills (used or available), number of RR remaining for the two coaches, RR for the turn (used or available), score, ball location, unique action blitz (played, available, or currently playing), unique pass action, hand off and foul availability (used or available), current player movement remaining, weather, apothecary (used, available, empty), the number of available bribes ($\{0, 1, 2\}$), the number of available wizards ($\{0, 1, 2\}$) and the roll result of the current action (at least for reroll decision).

Consequently, the cartesian product of P and V defines all the possible states, such that $S = P \times V$. But even expressing a specific $p \in P$ MDP$_p$ such that $S_p = p \times V$, the space state is huge. As an illustration regarding the size of S, let's consider only the available positions of 22 different players on the pitch. The number of states of such a sub-problem approximates A^{22}_{390}, roughly equal to 5×10^{56}.

B. Action modeling

In order to define the set of actions A, we first identify all the primitive actions $\mathcal{P}$ available during the main phase. We classified them according to whether or not they are directly dependent of a dice roll, and whether or not these actions are forced (*i.e.* they must be performed when available). The Fig. 3 shows these primitive actions classified. By composing these primitive actions, we can produce actions that can be mapped to the common actions decided by the coaches. In this paper, we intentionally skip specific action skills like *throw team mate*, *hypnotic gaze*, etc.

In the rest of the paper, primitive actions are written in lower case, and composite actions are written in upper case (*e.g.* BLITZ), for the sake of readability.

1) Actions not depending on a dice roll, $\mathcal{D}^-$:

Activate a player: A coach selects which next player to play. This action automatically terminates the activation of the previous player and it cannot be played again until the next turn. After being activated, a player movement point mp depends on its movement attribute such that $mp = Mv$.

Declare a BLITZ action: Only one BLITZ action can be performed during a coach turn. It must be declared just after activating a player. Note that BLITZ is not a primitive action and it will be further explained.

Move a player of 1 square: The activated player goes to a free adjacent square. Note that if the player moves from a square adjacent to an opponent player, then a dice roll is required to determine whether or not this action succeed (RR are applicable). This action costs 1 mp.

Foul: It consists in committing a foul on an adjacent opponent player. Note that the foul outcome is determined by a dice roll. The outcomes are: nothing happens, the opponent is stunned, the opponent is injured and/or the fouling player can be whistled by the referee.

Block: It consists to block an adjacent opponent player. The success of a block is determined by a dice roll.

Pass (or throw) the ball: Only one pass action can be performed during a coach turn. The ball is passed to another player, irrespective of their team, or the ball is thrown on a free square. Either way, the range is conditioned by a distance limit. The action success is determined by a dice roll *if* no successful interception has been made.

Intercept the ball: Action available when a player is between the thrower and the destination. The outcome is determined by a dice roll.

Stand up: Action to stand up a knocked down player. The outcome depends both on a dice roll and on the player's movement attribute. If $Mv \geq 3$, stand up action always succeed. It costs 3 mp.

Hand off the ball: Only one hand off action can be performed during a coach turn. The ball is given to an adjacent player (cannot be an opposing player). This action implies that the active player has the ball.

Pick up the ball: A forced action occurring when a player move to a square containing the dropped ball. Note that the success is determined by a dice roll.

Catch the ball: A forced action occurring when the ball lands on a player square, irrespective of the player's team. The success is determined by a dice roll.

Score a Touchdown: A forced action occurring when a player with the ball crosses the goal line of the other team.

2) Actions depending on dice roll, $\mathcal{D}^+$:

Select a dice roll result: The coach chooses the dice –thus its result– from of a roll of several block dice (*e.g.* 3DB).

Reroll: The coach re-rolls a dice roll using a RR (thanks to a skill or by using a general one).

Following a block: When a block result is pushing the opposing player, the initiating player can go to the square previously occupied by the pushed opponent.

Use apothecary: Some actions can injure players. When a player is being injured, the coach can decide to heal it by using

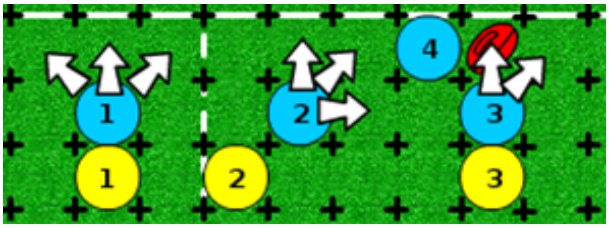

Fig. 4. Basic push actions. Yellow players ($\mathcal{Y}_p$) push the blue ones ($\mathcal{B}_p$).

an available apothecary. The outcome depends on a dice roll.
Use a bribe: Bribe action can be performed when the referee calls for a fault of the player who committed the foul. Depending on the success of the dice roll, the player may not be excluded from the match.
Use of specific skills: Some skills can only be performed during outcomes of specific actions, and are conditioned by dice rolls. A coach can choose to use them or not (some can only be used once per turn).
Push player(s): When a block action results in pushing a player, the coach of the player who blocks has to choose the push direction between one of the three free squares behind the pushed player. If no free square is available, the pushed player can be pushed toward a non-free square, recursively cascading the push mechanism until a player is pushed on a free square, as shown in Fig. 4 and Fig. 5. This can be modeled by transposing the pushed player of the pushing phase n into the pushing player of the $n+1$ pushing phase.
Additional blocking: When a player performs a push –while it has the *frenzy* skill– it is forced to follow the block, then to perform another block action on the same pushed player.

We classify all these primitive actions $\mathcal{P}$ as $\mathcal{P} = \mathcal{D}^- \cup \mathcal{D}^+$. Interestingly, this provides some space reductions and planning optimizations, as discussed in the section IV.

Now that primitive actions has been presented and classified, they can be composed in order to be mapped to more composite actions available during the game.

3) Composite actions, $\mathcal{G}$:

A composite action $g \in \mathcal{G}$ can be seen as a partially ordered set representing the succession of several primitive actions $p \in \mathcal{P}$, such that:

$$\forall g \in \mathcal{G}, g = (\mathcal{X}, \leq), \mathcal{X} = \{p_i \mid p_i \in \mathcal{P}\} \quad (1)$$

where the order of the elements in the set indicates the moment a primitive action has to be performed relative to the others on the set. Performing a composite action with a player automatically activates this player.

For the sake of readability, composite actions are described below using a *regex* format and written in uppercase. Note that this is not conventional and it is only done to provide a brief overview of the primitive actions composition.

MOVE $\Leftrightarrow$ (stand up?) (move*): Stand up a knocked down player and perform up to X primitive move actions, where X depends on Mv.

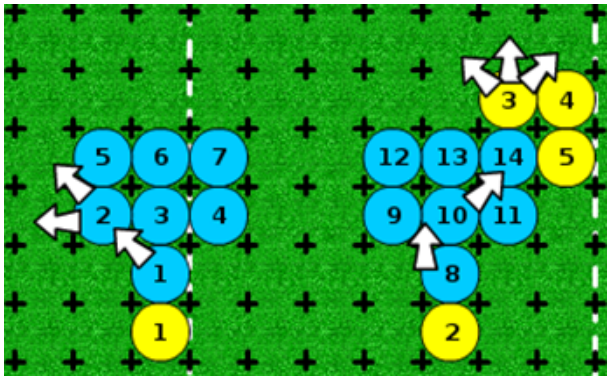

Fig. 5. Complex push actions examples. To the left: $\mathcal{Y}_1$ chooses to push $\mathcal{B}_1$ on $\mathcal{B}_2$, it can then be pushed on one of the two free squares behind it. To the right: illustration of a recursive push made by $\mathcal{Y}_2$ to $\mathcal{B}_8$, $\mathcal{B}_8$ to $\mathcal{B}_{10}$, $\mathcal{B}_{10}$ to $\mathcal{B}_{14}$ then $\mathcal{B}_{14}$ to $\mathcal{Y}_3$.

BLOCK $\Leftrightarrow$ (block) (reroll?) (select dice) (following block?) (push player(s)*): Block an adjacent opponent player and do all prospective actions among reroll, select a block dice, following block (if available) and push players.
BLITZ $\Leftrightarrow$ (blitz declaration) (stand up?) (move*) (BLOCK?) (move*): BLITZ always starts with its declaration phase, then standing up the player –if needed–, followed by potentially moving the player to connect for a BLOCK before choosing to continue to move or not.
FOUL $\Leftrightarrow$ (MOVE?) (foul): Performing a foul on a player may be preceded by a composite MOVE action. The foul action ends the turn of the performing player.
PASS $\Leftrightarrow$ (MOVE?) (pass): Performing a pass may be preceded by a composite MOVE action. Performing such a PASS ends the turn of the player.
HAND OFF $\Leftrightarrow$ (MOVE?) (hand off): Performing a hand-off action may be preceded by a composite MOVE action. This action ends the player turn.

4) Transition functions, T:

The *Blood Bowl*'s rules provide the necessary context for defining the transition functions T of the MDP, which we define accordingly. Incidentally, the number of transitions $|T|$ is in the same order of magnitude as $|S|$. Nonetheless, despite being a large set, each transition function only has a local impact on the game and therefore does not change the state of the game excessively. For instance, a free move just changes the player position to one of the eight squares around it.

Consequently, we will only illustrate some of the main game's transitions through some examples, illustrated from Fig. 6 to Fig. 12. Note that from the Fig. 7 knock down, stun and injury appear. We symbolized knock down by a white "/" on the afflicted player, stunned player by a white "X" and injured player by a white skull.

In Figs. 7, 8 and 10, when a player is knocked down or fouled, an *armor roll* must be performed (RR is impossible here). If the roll is successful (*i.e.* the armor roll is higher than the player's armor attribute) then an *injury roll* is needed.

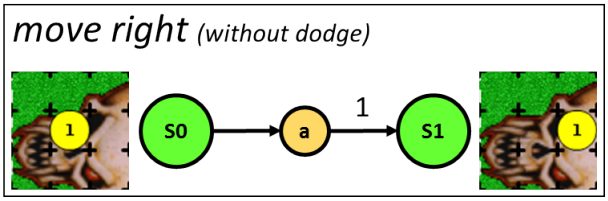

Fig. 6. Basic move right action. From the state S0, $\mathcal{Y}_1$ goes to the state S1 with a probability of 1, that is moving one square to the right.

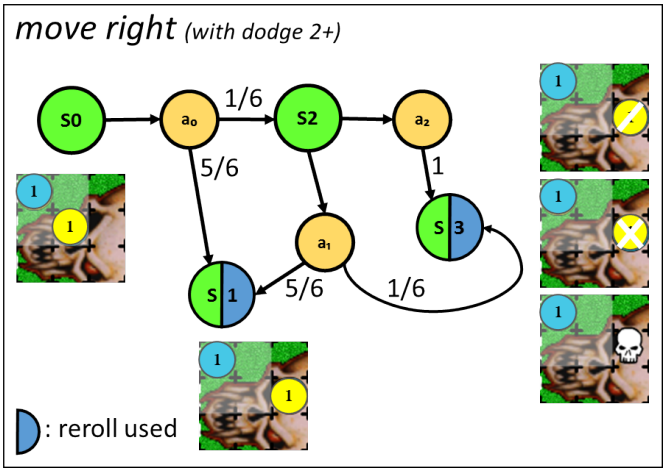

Fig. 7. Complex move right action with a 2+ dodge. RR is available. $\mathcal{Y}_1$ moves to the right and needs to dodge $\mathcal{B}_1$. From S0, by doing a_0, $\mathcal{Y}_1$ has a $5/6$ probability to succeed and being in S1, and a $1/6$ probability to fail and being in S2. From there, the coach can use its RR (a_1) or not (a_2). The former takes $\mathcal{Y}_1$ to either the state S1 or S3, respectively, with a $5/6$ probability and a $1/6$ probability. The later takes $\mathcal{Y}_1$ to the S3 – where we have simplified the KO/stun/injury outcome.

Depending on the outcome, the nature of the injury is: *stunned* (injury roll score $i < 8$); *KO* ($8 \leq i \leq 9$); *injured* ($i > 9$). In a *KO* or *injured* situation, the concerned coach can use an apothecary to alleviate the player status: a player *KO* will then become *stunned*, and *injured* player will be put in the coach reserve and made available for the next run.

Notwithstanding that we only use primitive actions in those examples, transition functions can involve composite actions as well. It is therefore apparent that the full size of the related MDP will be very important.

IV. Towards a Multi Layered Framework

As the previous section clearly shows, computing an optimal policy for the entire main phase of *Blood Bowl* – or any other game with very high branching factor – is currently intractable. Thus, planning proficiency becomes essential to perform well. *Blood Bowl* complexity has helped us to highlight an opponent-wise hierarchy of planning, with a growing complexity. In this section, we present a nomenclature of these levels and show where the current state of the art algorithms for *Blood Bowl* fall in. We also discuss introducing expert knowledge in the framework in a way that could benefit models for these kind of games and foster better AI algorithms.

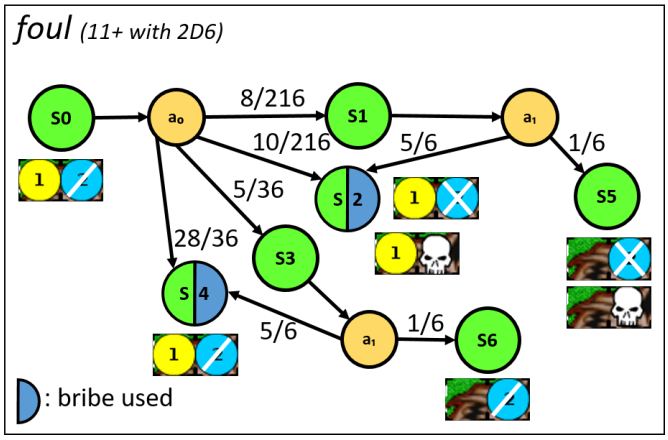

Fig. 8. 11+ Foul action. Yellow coach has a bribe. $\mathcal{Y}_1$ does a foul action a_0 on $\mathcal{B}_2$, currently knocked down, and can be in one of the four states S1, S2, S3 or S4. In S1, armor roll is successful, but the referee calls out a fault on $\mathcal{Y}_1$ with a probability of $8/216$ (twice 6 or a 5 and a 6 for the armor rolls, then a double for the injury rolls ($1/6$), thus $p = 1/36 + 2/36 \times 1/6 = 8/216$). In S2, armor roll is successful and the referee sees nothing with a probability of $10/216$ (5 and 6 for the armor rolls and no double for the injury rolls ($5/6$)). In S3, armor rolls fail and the referee calls out a fault on $\mathcal{Y}_1$ with a probability of $5/36$ (each double lower than 11). In state S4 (as well as S6), the foul fails and the referee sees nothing. When a fault is called, the coach can use a bribe a_1, with a probability of $5/6$ of being graciously accepted by the referee: if the foul roll succeeds, the injury roll determines what happens.

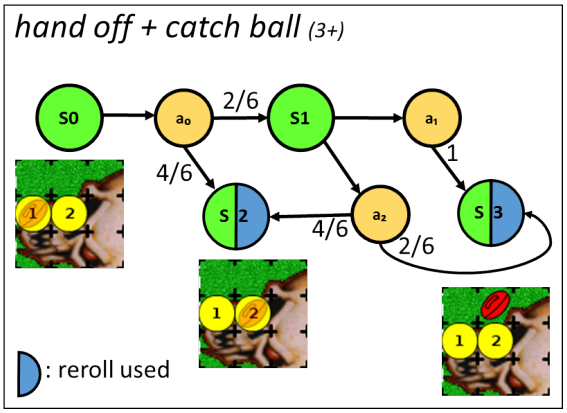

Fig. 9. Hand off, then 3+ catch ball actions. RR is available. $\mathcal{Y}_1$ hand off automatically succeed, then $\mathcal{Y}_2$ catch action a_0 has a $4/6$ probability to succeed, and a $2/6$ probability to fail. Regarding the later, the coach can use its RR. In S2, $\mathcal{Y}_2$ has the ball, while in S3 the ball is missed and bounces one square away –including to another player who can do a catch action.

A. A nomenclature of the planning level in Blood Bowl

Level 1 – One player At this level of planning, each player's actions are planned independently of the other players, until all the players have depleted their actions – or a turnover is called. In this plan, the best action is always played, irrespective to the player and without any consideration regarding the next player action. In other words, it consists of finding, locally, the best outcome at a specific time for a player. The main difficulty of such an approach is to determine the quality of the action.

This kind of approach is implemented in *GrodBot* and

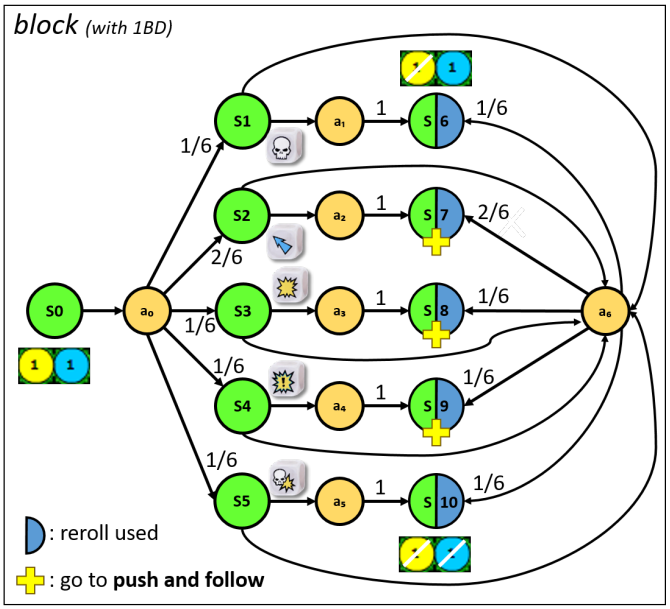

Fig. 10. Block action. $\mathcal{Y}_1$ performs a 1DB block (a_0) on $\mathcal{B}_1$. The side of the dice dictates the outcome of the block (cf. Fig. 2). Note that a coach can perform a reroll action a_6 to modify the initial outcome. S7, S8 and S9 are further described in Fig. 11

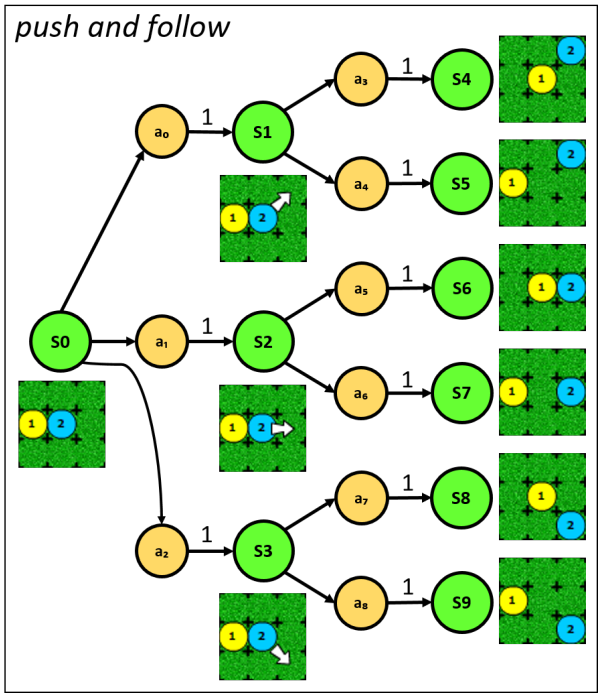

Fig. 11. Push and follow actions. After a block, in some cases, blocking player $\mathcal{Y}_1$ can push the blocked $\mathcal{B}_2$. Actions a_0, a_1 and a_2 correspond to the three push directions. In states S1, S2 and S3, blocking player can choose to follow the block (a_3, a_5, a_7) or not (a_4, a_6, a_8), resulting the six states from S4 to S9.

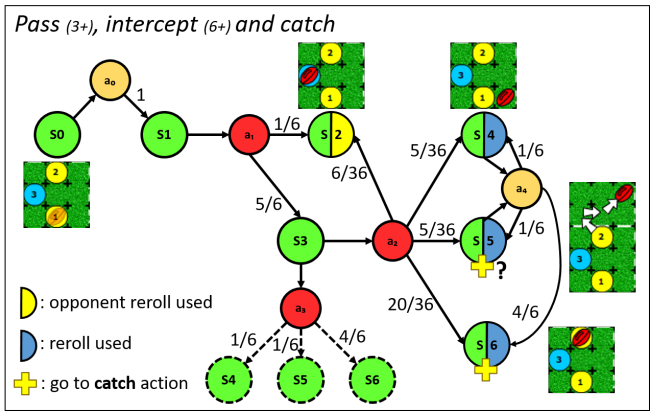

Fig. 12. 3+ pass with a 6+ interception action. In S0, $\mathcal{Y}_1$ declares a pass (a_0). Immediately after, the opposing coach can try to intercept it (a_1) with $\mathcal{B}_3$. The ball can be intercepted with a $1/6$ probability, or not thus being in S3 with a $5/6$ probability. In the later, the opposing coach can use a reroll action (a_2). If after using RR the interception fails again, the pass is made. Its quality is determined by a pass roll: clumsy (S4), inaccurate (S5) or perfect (S6). The yellow coach can use its RR here (a_4). Note how the probability of the outcomes of the pass change drastically if the opposing coach uses its RR or not.

Minigrod [1], winners of *Bot Bowl I and II*[1]. So far and to our knowledge, *Minigrod* is the reference algorithm in AI for *Blood Bowl*. Its proficiency is somewhat equivalent to a beginner human player.

Level 2 – Multiple players At this level of planning, x players' actions are considered, such that $x = \prod_{i=1}^{n} |\mathcal{A}_{i,j}|$, where $\mathcal{A}_{i,j} \in A$ are the actions potentially available[2] for a player i of the jth team and $n \leq 11$. Thus, level 1 planning is a specific case where $n = 1$. Considering our modeling, it is clear that the more n increases, the more the complexity of the planning phase is significant. It is generally accepted that the average proficiency of a human coach for planning lies in this level. With $n \in [0; 4]$, finding some potential *good* actions (e.g. another player assists a block) can be tractable.

Level 3 – One turn

At this level of planning, all the actions available during a team turn are considered while searching a potential solution. Therefore, $x = \prod_{i=1}^{m} |\mathcal{A}_{i,1}|$, where m is the number of playable players for the jth team. This proficiency of planning concerns very good human coaches - even if their plans contain flaws. It is comparable to compress sub-spaces S' of our model into new states $\mathfrak{S}$, *s.t.* $S_2 = \mathfrak{S} \cup (S \backslash S')$, leading to a simplified MDP, *s.t.* $\langle S_2, A, T, R \rangle$.

Level 4 – One turn with the two opponents At this level of planning, the entire turn is planned according to both the available actions and what the opponent players can do. Therefore, $x = \prod_{j=1}^{2} \prod_{i=1}^{m} |\mathcal{A}_{i,j}|$, where $\mathcal{A}_{i,2}$ are the available actions that an opposing player from the team B will be able to perform after the team A turn. This level of planning is only achieved by expert human coaches.

Level 5 – More than one turn At this level of planning, plans can be built on multiple turns and up to an entire half-

[1] https://njustesen.github.io/ffai/

[2] The order in which the actions are played have an influence.

time interval. This type of planning is undoubtedly intractable for now, in regards to its branch factor. We do not believe any human coach capable of efficiently elaborating such vast planning. Therefore, any AI algorithm that falls under this section could claim supremacy over human coaches.

B. Benefits of a MDP modeling for level 2 and 3 planning

Using a MDP modeling to drive the planning process in *Blood Bowl* appears to be quite relevant, because it is possible to model the entire main phase of the game and the game rules mathematically. By composing primitive actions $\mathcal{P}$, the probability of any plan in such a model can then be computed. Incidentally, the quality of a plan can then be derived through these probabilities.

As shown in Fig. 13, 3 composite actions $\mathcal{G}$ are planned for the blue players. These actions can be decomposed into primitive ones, with their associated probability to succeed, leading to the overall probability of success of the plan (BLITZ, then MOVE, then PASS, then catch). Probability of success or failure regarding common action sequences could be computed beforehand and used as is. The Table I illustrates this for several actions.

Moreover, classifying all the primitive actions $\mathcal{P}$ as $\mathcal{P} = \mathcal{D}^- \cup \mathcal{D}^+$ brings interesting modeling properties, especially regarding optimization. It is possible to define a new MDP' such that $MDP' \subset MDP$, where $MDP' = \{S', \mathcal{D}', T, R\}$ (*e.g.* $S' = V$). With all that, the space search can be drastically reduced under certain circumstances or considerations.

For example, by only considering $\mathcal{D}^-$ actions $\{move, pass, touchdown, catch, handoff\} \cup \{reroll\}$, with $reroll \in \mathcal{D}^+$, we simplify the problem by considering only dice-independent actions, but we can still search for a potential sequence of actions leading to a *touchdown* in one turn with 3 players.

C. Simplify the prospect of actions

It is important to consider and explore failure when establishing a plan (*e.g* does players' position offer a good cover in case of a turnover?). That is because, in a failure case scenario, the turnover mechanism of *Blood Bowl* does not allow the coach to play anymore until its next turn. With this consideration in mind, some composite actions could be simplified under certain circumstances.

For the composite MOVE action, when a player moves, it has up to 8 possibilities (all the squares around it). It goes the same for the move action that follows this move. When the number of dodge actions is strictly less than 2, considering the path with the highest probability to succeed in the MDP to go to the objective square B from the starting square A should be interesting, otherwise (*e.g* 2 or more dodges) it is not always the best plan to go with. For example, while attempting a MOVE action including two 2+ dodges with a single RR, the probability that the action succeed is $25/27$: if the first dodge is successful, the second has a success probability of $35/36$. If this first dodge fails, then succeeds the second time thanks to the reroll action, the success of the second dodge decreases to $5/6$.

TABLE I
FAILURE PROBABILITY TABLE, AS SUGGESTED IN [10]

Dice rolls	Calculation	%
3BD(RR, block)	$(1/6)^6$	0.002
2BD(RR, block)	$(1/6)^4$	0.077
3BD(RR, no block)	$(1/3)^6$	0.137
2BD(RR, block)	$(1/6)^4$	0.077
3BD(RR, no block)	$(1/3)^6$	0.137
3DB(block)	$(1/6)^3$	0.463
2BD(RR, no block)	$(1/3)^4$	1.235
2+(RR) | 2BD(block)	$(1/6)^2$	2.778
3BD(no block)	$(1/3)^3$	3.704
2+2+(2RR)	$1-(35/36)^2$	5.478
2+2+(RR)	$1-(25/36+10/36*5/6)$	7.407
-2BD(RR, block)	$(1-(5/6)^2)^2$	9.535
3+(RR) | 2BD	$(1/3)^2$	11.111
2+2+2+(RR)	$1-((5/6)^3+3*1/6*(5/6)^3)$	13.194
2+ | 1BD(block)	$1/6$	16.667
2+(RR)2+	$1-35/36*5/6$	18.981
2+2+ (2RR, intercept 6+)	$1-(35/36)^2*5/6$ (	21.232
4+(RR)	$(1/2)^2$	25
-2BD(block) | 2+2+	$1-(5/(6)^2$	30.556
2+2+ (RR, intervept 6+)	$1-35/36*5/6*5/6$	32.485
3+ | 1BD	$1/3$	33.333
2+2+2+	$1-(5/6)^3$	42.13
5+(RR)	$(2/3)^2$	44.444
4+	$1/2$	50
-2BD	$1-(2/3)^2$	55.555
5+	$2/3$	66.666
6+(RR)	$(5/6)^2$	69.444
6+	$5/6$	83.333

D. Looking for failures and Chance impact

As the attentive reader may have noticed, dice rolls in Blood Bowl are abundant. Dice roll results have often an important effect on coaches' decision-making process. For example, coaches extensively rely on blocking specific players in their planning: regarding the blocking outcome, they either continue their initial plan or switch to another. Consequently, anticipating failure is a factor to be taken into account.

Incidentally, rerolls and how they are spent also have a high impact on the probabilities all along the entire game. In some situations, having a RR can change an unlikely successful action into a successful one. Additionally, some actions are safe ($P(p) = 1$) and it can be interesting to consider some of them before other risky ones regarding the turnover rule.

E. The quality criterion

The quality of a plan is conditioned by the quality of criteria used to evaluate potential solutions. These criteria can be defined through the use of utility functions. A utility function's design greatly depends on the type of algorithm that implements it. For instance, it can be fitness functions in evolutionary computing, or rewards in reinforcement learning, MDP, etc. [11]. Either way, that is the reason that we do not focus on reward functions in our modeling: designing them for high branching factor games is complex. Nonetheless, studying *Blood Bowl* has highlighted that introducing expert knowledge as variables feeding the model can efficiently reduce the space-complexity of the problem and drive these utility functions. For example, in the subsection IV-C, we consider stopping the sequence of move actions in case of a successful RR as

BLITZ						MOVE						PASS							
block (2D)	select dice	push & follow	move up	move up/right	move up/right	move up/left	move up	move up	move up	move up	move up/right	move left	move left	move left	move up/left	move up	move up	pass	catch
5/9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2/3	2/3

Fig. 13. Planning of the blue team (lizardmen) against the red team (chaos), made of 3 composite actions (19 ordered primitive actions) and one primitive action. The first action consists to BLITZ with a 2DB to succeed, then safely moves another player. Finally, with a third player, undertake a PASS action which begins by safe moves and ends by sequencing a 3+ pass action then a 3+ catch action. All risky actions ($p < 1$) are assumed to be successful. 2DB blocking has a probability of $5/9$ to KO the opponent, 3+ passing and 3+ catching both have $2/3$. Since all the moving actions are safe, the probability of the plan to succeed is $p = 20/81 = 5/9 \times (2/3)^2$ without RR.

an expert knowledge: injecting this rule into the model can reduce the complexity of the MOVE action.

V. Conclusion and Perspectives

In this paper, we studied the main phase of the game *Blood Bowl*, which has been recently introduced as a new board game challenge for AI. We propose to model *Blood Bowl* as a MDP, incidentally classifying players' actions and game states, and show how this model fosters planning prospects.

Our proposition is the first to model *Blood Bowl* as an MDP to our knowledge, and could serve as a first multi layered framework for future AI algorithms trying to tackle *Blood Bowl* challenges, or any games with high branching factors, probabilistic outcomes and rare scoring situations – like evolutionary approaches in multi-actions games [12] [13]. Such a framework could also be used in deep neural network: data could be fed to a network dedicated to model the game according to the first framework level, then using this model as another input in another network to model the game according to the next layer of the framework, and so on. Eventually, it will facilitate the participation to *Bot Bowl*, an AI competition about *Blood Bowl* using the *The Fantasy Football AI Framework*[3].

Meanwhile, we are currently working on defining a tactical framework for the game in order to bring expert knowledge inside our modeling and ultimately be able to use this knowledge directly inside algorithms and their core parts. We are also considering modeling other phases of the game, such as the *team building*, which is assimilable to a stochastic optimization problem. Moreover, with the release of the newest version of *Blood Bowl* (a.k.a. the 2^{nd} *Season Edition*), we have an exciting opportunity to study how a framework is susceptible to change when the rules of a game evolve.

[3]https://github.com/njustesen/ffai

References

[1] N. Justesen, L. M. Uth, C. Jakobsen, P. D. Moore, J. Togelius and S. Risi. "Blood Bowl: A New Board Game Challenge and Competition for AI", 2019 IEEE Conference on Games (CoG), London, United Kingdom, 2019, pp. 1-8.

[2] Burmeister, Jay, and Janet Wiles. "The challenge of Go as a domain for AI research: a comparison between Go and chess." Proceedings of Third Australian and New Zealand Conference on Intelligent Information Systems. ANZIIS-95. IEEE, 1995.

[3] Vaithyanathan, Shivakumar, and Byron Dom. "Generalized model selection for unsupervised learning in high dimensions." Advances in neural information processing systems. 2000.

[4] M. Kearns, Y. Mansour and A.Y. Ng. "A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes". Machine Learning 49, 193-208 (2002).

[5] L. Kocsis and C. Szepesvári, "Bandit Based Monte-Carlo Planning". Machine Learning: ECML 2006, pp. 282-293 (2006)

[6] J. Johnson. Blood bowl handbook: Blood bowl living rulebook6.0, 2016.

[7] S. Mohandas and M. A. Nizar, "A.I for Games with High Branching Factor," 2018 International CET Conference on Control, Communication, and Computing (IC4), Thiruvananthapuram, 2018, pp. 372-376.

[8] R. Bellman. A Markovian Decision Process. Journal of Mathematics and Mechanics, vol. 6, pages 679-684, 1957.

[9] M. Littman, T. Dean and L. Kaelbling. On the complexity of solving Markov decision problems. In 11th Conference on Uncertainty in Artificial Intelligence, pages 394-402, 1995.

[10] Wreckage. The Thousand Losses Playbook, https://www.dropbox.com/s/2ednwzorq89gc4g/1000lossesed1.pdf. Last access: 30 march 2020.

[11] G. N. Yannakakis and J. Togelius. Artificial Intelligence and Games. Springer, 2018. http://gameaibook.org.

[12] Justesen N., Mahlmann T., Togelius J. (2016) Online Evolution for Multi-action Adversarial Games. In: Squillero G., Burelli P. (eds) Applications of Evolutionary Computation. EvoApplications 2016. Lecture Notes in Computer Science, vol 9597. Springer, Cham

[13] H. Baier and P. I. Cowling, "Evolutionary MCTS for Multi-Action Adversarial Games," 2018 IEEE Conference on Computational Intelligence and Games (CIG), Maastricht, 2018, pp. 1-8.

[14] A. Gustafsson, 2019, Winner Prediction of Blood Bowl 2 Matches with Binary Classification, http://hdl.handle.net/2043/29895