A Markov Game Model for Evaluating National Hockey League Play-By-Play Events

Abstract

A variety of advanced statistics are used to evaluate player actions in the National Hockey League, but they fail to account for the context in which an action occurs or the long-term effects of an action. We present a novel approach to construct a multi-agent Markov Decision Process, also called a Markov Game model, from sequences of player actions that accounts for action context and cross-sequence influence, and supports reinforcement learning over a variety of objective functions to evaluate actions and players. A dynamic programming value iteration algorithm is used to learn the values of states in the Markov Decision Process and player actions. Player actions are found to have both positive and negative effects dependent on the context the action occurs in. Players are evaluated and ranked according to the computed values of their actions during matches. We examine values of context states with and without event sequences to observe the benefit of including event histories.

1 INTRODUCTION

As sports have entered the world of big data, it is increasingly important for teams to find new methods of evaluating players to gain insight into their players as well as their opponents. By making use of advanced statistics, player performance can be measured more accurately and future performance can be predicted. From the perspective of a general manager, advanced statistics could be used to effectively buy wins and increase the entertainment value of sports. Previous research has been performed to use the data available to generate advanced statistics for the National Hockey League (NHL), such as the Adjusted Plus-Minus [Macdonald, 2011], the Expected Goals Model [Macdonald, 2012], the Complete Plus-Minus [Spagnola, 2013], and the Total Hockey Rating (THoR) [Schuckers and Curro, 2013]. A few problems with these advanced statistics are that they ignore the context of actions within a game and lose interpretability as a result, and often have difficulty discerning between the actions or contributions of teammates. The predictive accuracy of these advanced statistics are often low, and do not form very informative features for classification problems [Weissbock and Inkpen, 2014]. A key challenge is to use the data available to analyze player actions within the context of a game and measure their contribution to their team such that the measurements are useful in both a predictive manner and for economic valuation of players.

In this paper, we propose a Markov Decision Process (MDP) construction algorithm to model all observed player actions within a variety of contexts in an ice hockey game, which can be used to measure the effect of player actions for various objectives. These quantifications for each action are derived using reinforcement learning in a dynamic programming value iteration algorithm.

1.1 TASK DESCRIPTION

A key problem with using advanced statistics to evaluate NHL players is that the context in which a player's action occurs is ignored, and some actions are ignored entirely. As a thought experiment, a goal that is scored to tie the game would be more useful to a team, with respect to wins, than scoring a goal when the team is already ahead by six goals. Since ice hockey is by nature a low-scoring game [Lock and Schuckers, 2009], it is intuitive that the former significantly increases a team's chance of winning, and the latter does not modify a team's chance of winning very much. Our method will take into account the context of the game for player actions, as well as the history of events that have occurred, to measure the value of player actions and provide more accurate computations of a player's contributions.

1.2 MOTIVATION

Recent publications have shown the importance of incorporating the context within a hockey game into predictions and valuations. Zone entry information was used in [Tulsky et al., 2013] to analyze player performance and learn effective strategies for entering the offensive zone. [Schuckers and Curro, 2013] also incorporated zone information as an adjustment for player contributions. Both zone and special team situations, such as powerplay or penalty kill situations where there is a manpower difference between teams, were modeled in [Macdonald, 2012]. Both of these context features, as well as goal differential, have an impact on scoring rates [Thomas et al., 2013], but these context features have not yet been combined into a single effective model to evaluate players.

Trees of event sequences have also been used extensively in games outside the sports domain. Applications range from turn-based board games, such as chess [Berliner, 1979] where tree search is frequently used to determine optimal actions, to the continuous domain, where real-time strategy games [Churchill and Buro, 2013] make use of trees of event sequences to facilitate search algorithms to determine the best actions. It is then intuitive to apply graphical models to continuous-flow sports, such as ice hockey, where each play consists of a sequence of events and actions. The graphical model used in our approach is a multi-agent Markov Decision Process, also called a Markov Game model.

This method will be useful for team coaches who want to pick players for certain contexts of the game. For example, each team in the NHL picks a few players for powerplay situations that they believe are likely to score a goal. By analyzing player contributions within powerplay situations as opposed to their general contributions, coaches would have better information on which players to choose in order to maximize their likelihood of scoring a powerplay goal. From an economic perspective, team general managers can make use of the evaluations returned by this model for enriched player analysis and improved player acquisitions.

1.3 APPROACH

The general approach is to map all observed NHL play-by-play events into a tree of events for each game context, where each play sequence forms a branch of the tree under each game context. Game context consists of goal differential, manpower differential, and period, three features that have not been previously examined together. To facilitate cross-sequence effects, an edge is added from the leaf node of a play sequence to the starting context state of the following play sequence, transforming the graph from a tree structure to a context-inclusive Markov Decision Process. Rewards for different objective functions, such as goals, and penalties, are then applied to each event node of the MDP, and the values of each event can be learned using value iteration as a reinforcement learning technique. These values are then used to quantify player contributions and aggregated over contexts, games, and seasons to get total player values.

1.4 EVALUATION

Our method is evaluated by firstly examining our calculated values for player actions and comparing our computations with previous valuations of player actions. Next, we apply these action values to each player as they perform the action in a match, aggregate these values to determine a player valuation, and compare player valuations to their salary. We also rank players in different contexts and compare against previous player rankings.

1.5 CONTRIBUTIONS

Our main contributions may be summarized as follows:

1. A scalable data structure for measuring player contributions over configurable objective functions,
2. The first application of reinforcement learning in a continuous-flow sports domain,
3. A new and intuitive method for analyzing player actions within the context of a game.

1.6 PAPER ORGANIZATION

We start by addressing related work in measuring player contributions and machine learning in sports in Section 2. We will then give some background information on NHL play-by-play sequences and notation used in our method in Section 3. The two algorithms for our method are then described in detail in Section 4.

2 RELATED WORK

[Lock and Schuckers, 2009] first determined the values of actions in the NHL by using the number of goals scored after a specific action in the next ten seconds over the number of all occurrences of the specific action. For penalties, the duration of the penalty was used as the lookahead window. This looked at only actions in the 2006/2007 NHL season. The purpose of this was to generate an adjusted plus-minus statistic based on the plus-minus statistic used in the NBA [Rosenbaum, 2004]. A drawback to this method is it assumes a fixed value for each action and does not account for the context in which an action occurs. It also gives a natural bias for actions to be valued in favor of a player's team or in favor of the opposing team. Our method not only examines a much larger dataset, but it will show that the values of action vary not only in magnitude, but also in how the action favors each team in different contexts.

[Gramacy et al., 2013] examined the value of NHL players by using regularized logistic regression with the players on ice when a goal occurred. This method made the assumption that player and team values are consistent over the four seasons analyzed. While it did use teams as a latent variable to adjust player contributions accordingly, it did not account for home ice advantage, as advocated by [Schuckers and Curro, 2013]. Logistic regression also does not account for the history of actions and player contributions as a result of previous actions. Our method takes in account the sequence of actions and their contribution to some reward such as a goal or a penalty, and can evaluate players and teams on a variety of metrics by altering the reward function for value iteration.

In [Schuckers and Curro, 2013], actions were evaluated based on whether or not a goal occurred in the following 20 seconds after an action. A drawback to this is that it assumes a fixed value for every action and does not account for the context in which an action takes place. Furthermore, the window of 20 seconds or to the end of the play sequence restricts the lookahead value of each action. Our method is not restricted to any particular time window, but takes into account the event history and looks ahead to the next goal, allowing greater flexibility and more accurate evaluation of player actions.

[Sidhu and Caffo, 2014] have examined reinforcement learning on a Markov Decision Process for pitching in Major League Baseball, and used reinforcement learning to find exploitable strategies for batters. Our work is similar as our method uses value iteration on a Markovian state space, but it differs in that it examines a much larger state space consisting of $1,325,852$ states, rather than the 12 states used by [Sidhu and Caffo, 2014]. Our model is also much more scalable, as it can be used for multiple objective functions, such as expected values or probabilities of goals or penalties. Their model is focused on determining an optimal policy for batting or pitching strategy. While our model easily facilitates finding an optimal playing policy, we focus on the values of each state for the purpose of evaluating players, rather than determining team strategies.

3 BACKGROUND, NOTATION AND DATA FORMAT

We will give a brief overview of rules of play in the National Hockey League. For detailed rules of play in the National Hockey League, refer to [National Hockey League, 2014]. NHL games consist of three periods, each 20 minutes in duration. A team will try to score more goals than their opponent within three periods in order to win the game. If the game is still tied after three periods, the teams will enter a fourth overtime period, where the first team to score a goal wins the game. If the game is still tied after overtime during the regular season, a shootout will commence. During the playoffs, overtime periods are repeated until a team scores a goal to win the game.

Teams have five skaters and one goalie on the ice during even strength situations. Teams can pull their goalie to have an additional player on the ice and gain a scoring advantage, with the empty net also increasing the risk of being scored on. Penalties result in a player sitting in the penalty box for two, four, or five minutes and the penalized team will be shorthanded, creating a manpower differential between the two teams.

3.1 NOTATION

The following notation is used to describe the state space in NHL play sequences and player actions, and is used in the MDP construction and Value Iteration algorithms. MDP notation follows [Russell and Norvig, 2010], and a modification of the notation used by [Littman, 1994] is used to describe the multi-agent setup specific to NHL games. Notation for value iteration follows [Mitchell, 1997].

1. Goal Differential GD is calculated as Number of Home Goals - Number of Away Goals. A positive goal differential means the home team is leading (away team is trailing), and a negative goal differential means the home team is trailing (away team is leading).

2. Manpower Differential MD is calculated as Number of Home Skaters on Ice - Number of Away Skaters on Ice. A positive manpower differential typically means the home team is on the powerplay (away team is penalized), and a negative manpower differential typically means the home team is shorthanded (away team is on the powerplay).

3. Period P represents the current period number the play sequence occurs in, typically ranging in value from 1 to 5. Periods 1 to 3 are the regular play of an ice hockey game, and periods 4 and onwards are for overtime and shootout periods as needed.

4. Zone Z represents the area of the ice rink in which an action takes place. Z can have values Offensive, Neutral, or Defensive, relative to the team peforming an action. For example, $Z =$Offensive zone relative to the home team is equivalent to $Z =$Defensive zone relative to the away team. We exclude zone from the context space, as it is not typically fixed throughout a play sequence, but include it as an attribute of player actions instead.

5. State s represents the play sequence consisting of action events $a_1, a_2, \ldots, a_n$ and with a particular GD,

MD, and P as the context. The play sequence may be empty, in which case state s is purely a context node. A successor state s' represents the play sequence consisting of action events $a_1, a_2, \ldots, a_n, a_{n+1}$ and a particular GD, MD, and P.

6. $R(s)$ is the reward value for each state s. This value is dependent on the objective being analyzed.

7. $Occ(s)$ is the number of occurrences of state s as observed in the play-by-play data.

8. $Occ(s, s')$ is the number of occurrences of state s being immediately followed by state s' as observed in the play-by-play data. (s, s') forms an edge of the Markov Game model.

9. The transition function T is a 1-to-1 mapping of tuple $s \times a \rightarrow s'$. In this restricted multi-agent environment, only one team can perform an action from each state, although not in a turn-based sequence. For example, if a_{home} =FACEOFF, then a_{away} =NO OPERATION, and a_{home} is recorded as a.

10. The transition probability function TP is a mapping of $T \rightarrow \mathbb{R}$, and is computed as $\frac{Occ(s, s')}{Occ(s)}$.

11. $Q_i(s)$ is the value iteration function value for state s during iteration i. Note that $Q_0(s) = 0$ as an initialization step for the value iteration algorithm.

12. $impact(a)$ is the value of an action, which we compute to evaluate players.

3.2 DATA FORMAT

The NHL provides information about sequences of play-by-play events, which are scraped from `http://www.nhl.com` and stored in a relational database. The real-world dataset is formed from $2,827,467$ play-by-play events recorded by the NHL for the complete 2007-2014 seasons, regular season and playoff games, and the first 512 games of the 2014-2015 regular season. A breakdown of this dataset is shown in Table 1. Note that there are only 30 teams in the NHL, but some teams moved, so there are 32 teams in our dataset. For each event, the current goal differential GD, manpower differential MD, and period P are scraped from the play-by-play data. The type of events recorded by the NHL from the 2007-2008 regular season and onwards are listed in Table 2. Note that the top events are action events performed by players and the bottom events are start and end markers for each play sequence. Every event is marked with a timestamp and a zone Z. We include the zone information for each action-event, but an effective use of this temporal information is left as future work. Every action event is also marked with which team, Home or Away, carries out the action. An **action** is therefore a pair (Team, Action-Event). An event history is a sequence of **events** with at most one start and end marker. A **state** contains the current context of the match, which includes the goal differential GD, manpower differential MD, and period P, as well as the event-history at the time of an action. Potentially, there are $(18 \times 7 \times 7) = 882$ context states and $(7 \times 2 \times 3)^{40} = 42^{40}$ event histories. In the Markov Game Model formed from our dataset of observed sequences, there are 450 observed context states, and the Markov Game Model contains $1,325,852$ nodes of states and events. A sample of the NHL play-by-play data is shown in Table 3.

Table 1: Size of Dataset

Number of Teams	32
Number of Players	1,951
Number of Games	9,220
Number of Sequences	590,924
Number of Events	2,827,467

Table 2: NHL Play-By-Play Events Recorded

Event
Faceoff
Shot
Missed Shot
Blocked Shot
Takeaway
Giveaway
Hit
Goal
Penalty
Stoppage
Period Start
Period End
Shootout Completed
Game End
Game Off
Early Intermission Start
Early Intermission End

4 MODEL CONSTRUCTION AND REINFORCEMENT LEARNING ALGORITHMS

We first give an informal description of the MDP construction and reinforcement learning algorithms in Section 4.1. Next, we give a formal outline of the context-inclusive MDP construction algorithm using the NHL play-by-play data in Section 4.2. Lastly, we give a formal outline of the dynamic programming value iteration algorithm in Section 4.3.

Table 3: Sample Play-By-Play Data

GameId	Period	Event Number	Event
1	1	1	PERIOD START
1	1	2	faceoff(Home,Neutral)
1	1	3	hit(Away,Neutral)
1	1	4	takeaway(Home,Defensive)
1	1	5	missed_shot(Away,Offensive)
1	1	6	shot(Away,Offensive)
1	1	7	giveaway(Away,Defensive)
1	1	8	takeaway(Home,Offensive)
1	1	9	missed_shot(Away,Offensive)
1	1	10	goal(Home,Offensive)
…			

4.1 INFORMAL DESCRIPTION, INTUITION

Plays in the NHL form natural sequences of actions, typically starting with a faceoff and ending with a goal, penalty, or play stoppage. These events can be viewed as a choice of actions by each team. It is intuitive to then transform these sequence of events into a tree of events, or a game tree, where each subsequent event in a sequence is the child of the preceding event. We are also taking into account the context in which a play sequence occurs, so the tree must include the starting context of each play sequence as a state. The graph construction is performed as follows: the tree begins with a root state, or root node, where there is no context or sequence information. This followed by the node representing the context of the game the play is starting in. Then the sequence of events follows below the context node, with branches forming as different events occur over multiple sequences. The process is repeated for each new play sequence by starting from the root node and adding new states, or nodes in the graph, as new action sequences are observed. Actions, such as penalties, often have an effect on the following sequences of actions. In order to propagate these effects, an edge is added from each leaf node of a play sequence, which represents a play sequence ending with a goal, penalty, or stoppage, to the context state node of the following play sequence. This transforms the graphical model from a tree into a multi-agent Markov Decision Process called a Markov Game Model. For an in-depth explanation of Markov Decision Processes, refer to [Russell and Norvig, 2010]. For more details on Markov Game Models, refer to [Littman, 1994].

The next step is to perform reinforcement learning on the Markov Game Model, which will yield valuations of player actions in different context states. A dynamic programming value iteration algorithm is used as the reinforcement learning technique to determine the value of each node. The evaluation can be performed over many objective functions simultaneously, and is run iteratively until a convergence criterion is met or a maximum number of iterations is reached. For our experiments, we used nine objective functions shown in Table 4. Conditional Probabilities for goal scoring and receiving a penalty can also be derived by combining the probabilistic objectives for the home team and away team. These node values are then used to compute action values specific to each node, as the action values are dependent on the context and event history. As players perform these actions in a match, the action values are applied to the players and are used to evaluate player performance.

Table 4: Value Iteration Functions

Expected Goals
Probability that the Home Team Scores the Next Goal
Probability that the Away Team Scores the Next Goal
Expected Penalties
Probability that the Home Team Receives the Next Penalty
Probability that the Away Team Receives the Next Penalty

4.2 CONTEXT-INCLUSIVE MARKOV GAME MODEL CONSTRUCTION

The Context-Inclusive Markov Game Model Construction Algorithm is outlined in Algorithm 1. Note that the root is an empty node with no context or event information, and acts as a graph initialization. For each node, the context information GD, MD, and P are set when the new node is created, and the new action a is added to the sequence along with the zone Z that a occurs in. The reward $R(s)$ is also applied to each node as it is created, and the value of $R(s)$ is dependent on the objective function being used. The node counts $Occ(s)$ and edge counts $Occ(s, s')$ are recorded and applied to each node and edge respectively, and are used to generate transition probabilities TP for the value iteration in a maximum likelihood estimation. The function $incrementCount(s)$ is used to set node count $Occ(s)$, and $incrementCount(s, s')$ is used to set edge count $Occ(s, s')$. The NHL play-by-play event data records goals that occur, but do not record a separate event for the shot leading to the goal. Following [Schuckers and Curro, 2013], we record the shot leading to the goal in addition to the goal itself. This is done by injecting a shot event into the event sequence prior to the goal.

4.3 VALUE ITERATION DYNAMIC PROGRAMMING ALGORITHM

Now that the Markov Game Model for the play-by-pay events has been generated, the next step is to run a dynamic programming algorithm for value iteration to extract the values of different actions in different states. For an in-depth explanation of value iteration for reinforcement learning, refer to [Mitchell, 1997]. We use an undiscounted Q-function for our value iteration, following [Schwartz, 1993]. For expected values of goals or penalties, Equa-

Algorithm 1 Context-Inclusive Markov Game Model Construction

```
Require: NHL play-by-play data
 1: $root = new\ Node(empty)$
 2: for all games $g$ do
 3:     $current = root$
 4:     $previous = null$
 5:     $lastLeaf = false$
 6:     for all events $i$ in game $g$ do
 7:         if $current == root$ then
 8:             $incrementCount(root)$
 9:             $state = i.getStateInformation$
10:             if not $root.hasChild(state)$ then
11:                 $root.addChild(state)$
12:             end if
13:             $current = state$
14:             $incrementCount(current)$
15:             $incrementCount(root, current)$
16:             if $lastLeaf == true$ then
17:                 if not $previous.hasChild(current)$ then
18:                     $previous.addChild(current)$
19:                 end if
20:                 $incrementCount(previous, current)$
21:                 $lastLeaf = false$
22:             end if
23:         end if
24:         if $i ==$GOAL then
25:             $shotEvent = new\ Node(i, \text{“}SHOT\text{”})$
26:             if not $current.hasChild(shotEvent)$ then
27:                 $current.addChild(shotEvent)$
28:             end if
29:             $incrementCount(current, shotEvent)$
30:             $incrementCount(shotEvent)$
31:             $previous = current$
32:             $current = shotEvent$
33:         end if
34:         $event = new\ Node(i)$
35:         if not $current.hasChild(event)$ then
36:             $current.addChild(event)$
37:         end if
38:         $incrementCount(current, event)$
39:         $incrementCount(event)$
40:         $previous = current$
41:         $current = event$
42:         if $current.isTerminalEvent()$ then
43:             $lastLeaf = true$
44:             $previous = current$
45:             $current = root$
46:         end if
47:     end for
48: end for
```

tion 1 is used as the value iteration function. $R(s)$ is initialized based on the event being analyzed as an objective. For example, if the objective is to find the expected goals, $R(s) = 1$ when s is a Home Goal event, $R(s) = -1$ when s is an Away Goal event, and $R(s) = 0$ for all other events and states. This is done similarly for when using penalties as the objective. Note that $\frac{Occ(s,s')}{Occ(s)}$ forms the transition probability from state s to state s', but $\frac{1}{Occ(s)}$ is factored out to the front of the summation to speed computation time.

For the probability of the next goal, or next penalty, Equation 2 is used as the value iteration function. Here, EVENT can be Goal or Penalty, and TEAM can be Home or Away. For example, if you want to find the probability of the next home goal, then the EVENT would be Goal and TEAM would be Home. All events of type of EVENT are excluded from the first summation in Equation 2. This facilitates backing up the value 0 for the opposite TEAM. For example, if EVENT is Goal and TEAM is Home, TEAM:EVENT = Away:Goal is excluded from the summation, equivalent to backing up 0 for Away:Goal.

Once the objective functions are defined, the Value Iteration Dynamic Programming Algorithm can be executed on the MDP to determine the values of each state. The steps are outlined in Algorithm 2. Algorithm 2 shows the algorithm with the Expected Goals or Expected Penalties computation shown in Equation 1. To calculate other objectives, the calculation for $Q_{i+1}(s)$ can be replaced with Equation 2.

$$Q_{i+1}(s) = R(s) + \frac{1}{Occ(s)} \sum_{(s,s') \in E} (Occ(s,s') \times Q_i(s')) \tag{1}$$

$$\begin{aligned} Q_{i+1}(s) = &\\ &\frac{1}{Occ(s)}\Big(\Big(\sum_{\substack{(s,s')\in E \\ s' \neq EVENT}} \Big(Occ(s,s') \times Q_i(s')\Big)\Big) \\ &+ \Big(\sum_{\substack{(s,s')\in E \\ s' = TEAM:EVENT}} \Big(Occ(s,s') \times 1\Big)\Big)\Big) \end{aligned} \tag{2}$$

4.4 EXAMPLE

We will first give a small example of the context-inclusive Markov Decision Process construction. This will be followed by a few iterations of the value iteration dynamic programming algorithm.

Algorithm 2 Value Iteration Dynamic Programming Algorithm

```
Require: MDP, convergence criterion $c$, maximum number of iterations $M$
 1: $lastValue = 0$
 2: $currentValue = 0$
 3: $converged = false$
 4: for $i = 1; i \leq M; i \leftarrow i + 1$ do
 5:   for all states $s$ in the MDP do
 6:     if $converged == false$ then
 7:       $Q_{i+1}(s) = R(s) + \frac{1}{Occ(s)} \sum_{(s,s') \in E} (Occ(s, s') \times Q_i(s'))$
 8:       $currentValue = currentValue + |Q_{i+1}(s)|$
 9:     end if
10:   end for
11:   if $converged == false$ then
12:     if $\frac{currentValue - lastValue}{currentValue} < c$ then
13:       $converged = true$
14:     end if
15:   end if
16:   $lastValue = currentValue$
17:   $currentValue = 0$
18: end for
```

4.5 DISCUSSION

The context state space consists of goal differential, manpower differential, and period as context variables. Goal differential, manpower differential, and period typically remain fixed during a play sequence. The exception to this rule is that manpower differential can sometimes change during a play sequence when the goalie is pulled or a penalized player returns to the ice. The zone was left out context state space in order to decrease the number of context subtrees, which also reduces the number of nodes. Instead, zone is included in the action-event.

5 HARDWARE, EVALUATION AND RESULTS

The hardware used in our data collection, model construction, and value iteration computation are summarized. For evaluating our method, we first examine the information from the play-by-play data by examining the context contingency table without including sequence history. Next, we discuss the benefit of including cross-sequence loops in the Markov Game Model. We then use the Markov Game Model with cross-sequence loops to determine the values of each action, and compare with action values computed in previous works. Finally, we use our action values derived from the value iteration computation to score and rank NHL players, and compare player scores with salaries.

5.1 HARDWARE

NHL play-by-play data was obtained from `http://www.nhl.com` using the Selenium WebDriver with Python 2.7.6 [Salunke, 2014] on a 64-bit Ubuntu 14.0.4 LTS Virtual Machine with 4.8GB RAM and an Intel Core i7-2670QM CPU @ 2.20GHz $\times$ 8. Markov Game Model construction and value iteration computation was performed using Java Version 8 Update 25 on 64-bit Windows 7 Home Premium with 12GB RAM and an Intel Core i7-2670QM CPU @ 2.20GHz $\times$ 8.

5.2 STATE STATISTICS AND VALUES: NO SEQUENCES

general strategy: winning chance changes with goal scoring (goal diff). Goal scoring changes with penalties. Dtree can tell you how to achieve goals or penalities locally. State backup integrates these numbers into a single one, for both goal scoring and penalty achievement. (Player ranking?)

1. 24.1% of all Play-by-Play sequences end in either a goal or a penalty.
   (a) 8.9% of all Play-by-Play sequences end in a goal.
   (b) 15.2% of all Play-by-Play sequences end in a penalty.
2. Goals are more frequent than penalties only in the 4th period.
3. If a goal is scored on the powerplay, it is 76.2% likely to be a powerplay goal and 23.8% likely to be a shorthanded goal.
4. Gaining a powerplay significantly increases the conditional probability of scoring a goal.
   (a) If the away team is on the powerplay, they can be up to 55% more likely to score the next goal.
   (b) If the home team is on the powerplay, they can be up to 65% more likely to score the next goal.
5. Gaining a powerplay also significantly increases the conditional probability of receiving a penalty.
   (a) When the home team goes on the powerplay in Period 1, the conditional probability of the home team receiving a penalty jumps from 48.9% to 65.9%.
   (b) When the away team goes on the powerplay in Period 1, the conditional probability of the away team receiving a penalty jumps from 51.1% to 72.3%.
6. Scoring a goal significantly increases the probability of winning.

(a) When the home team scores a goal in Period 2 for a one goal lead, their probability of winning increases from 53.8% to 72.5%.

(b) If the home team scores another goal in Period 2 for a two goal lead, the probability of winning increases further to 86.5%.

(c) When the away team scores a goal in Period 2 for a one goal lead, their probability of winning increases from 46.2% to 66.6%.

(d) If the away team scores another goal in Period 2 for a two goal lead, the probability of winning increases further to 84.0%.

insert contingency table of sequences based on (goal differential, manpower differential, period). with the following fields:

sequence count. ends-in-goal-count. Home/Away. ends-in-penalty-count. Home/Away. Value for different reward functions. Computed with sequences in states. 1) Reward = win. 2) reward = goal. 3) reward = next goal.

5.2.1 State Contingency Table

The data are for the complete 2007-2014 seasons, as well as the first 512 games of the 2014-2015 season, and includes both regular season and playoff games. Table 5 includes statistics for the top-20 context states over all sequences. Note that positive differences are for the home team and negative differences are for the away team. For example, a Goal Difference of 7.1% means the home team is 7.1% more likely to score a goal in that context state than the away team. Similarly, a Penalty Difference of -33.2% means the away team is 33.2% more likely to receive a penalty in that context state than the home team.

Table 5 is formed from $590,924$ total sequences containing $52,793$ total goals and $89,612$ total penalties. Period 4 is sudden-death overtime, where it can be observed that this is the only period where goals are more frequent than penalties.

5.2.2 Sequences

Table 6: Event Sequence Statistics

Sequence Lengths	**Maximum**	**Average**	**Variance**
Overall	42	4.87	10.95
Sequence ends in a goal	38	5.85	9.66
Sequence ends in a penalty	42	4.10	10.92

Sequences ending in a goal tend to be longer in length, as also observed by [Thomas et al., 2013], and, on average, consist of 5.85 events.

5.3 SEQUENCE HISTORY AND CROSS-SEQUENCE LOOPING

[NOTE: Show change in action values with only local sequences, penalty loops, and full loops]

5.4 ACTION VALUES

We first compare the values computed for each action from our value iteration algorithm with the fixed values computed by Lock and Schuckers [Lock and Schuckers, 2009; Schuckers et al., 2011]. The action values used in Figure 1 are obtained using the Probability of the Next Home Goal as an objective function in our value iteration algorithm. Positive values are in favor of the a player's team, and negative values are in favor of their opponent. The values computed for each action in [Lock and Schuckers, 2009] are displayed as an asterisk in Figure 1. It is clear from Figure 1 that accounting for context and event history when performing an action affects the value of an action. All actions, with the exception of faceoffs won in the offensive zone, have at least one observance where the action is either a positive action or a negative action. The action values found in [Lock and Schuckers, 2009] tend to agree with the median of our action values. The exceptions to this are for blocked shots, faceoffs won in the offensive zone, penalties, and shots, but the postive/negative team bias found in [Lock and Schuckers, 2009] for each agrees with our medians. For example, penalties are known to generally be good for the opposing team, and shots are good for the player's team.

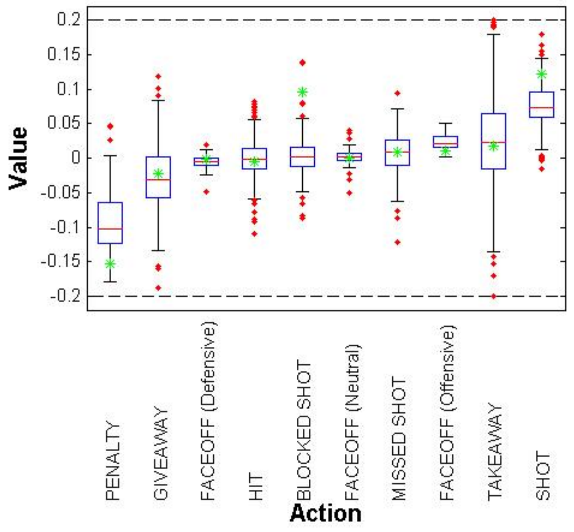

Figure 1: Action Values

Table 5: Statistics for Top-20 Context States

Goal Differential	Manpower Differential	Period	Number of Sequences	Winning Difference	Number of Goals	Goal Difference	Number of Penalties	Penalty Difference
0	0	1	78,118	9.7%	5,524	7.1%	11,398	-2.3%
0	0	2	38,315	7.6%	2,935	7.6%	5,968	-2.9%
0	0	3	30,142	2.9%	2,050	5.9%	3,149	-2.2%
1	0	2	29,662	45.1%	2,329	2.0%	4,749	2.2%
1	0	3	25,780	60.6%	2,076	4.3%	3,025	3.5%
-1	0	2	25,498	-33.2%	1,970	8.6%	4,044	-8.7%
1	0	1	24,721	41.5%	1,656	5.3%	4,061	3.4%
-1	0	3	22,535	-54.5%	1,751	0.7%	2,565	-18.3%
-1	0	1	20,813	-26.1%	1,444	4.6%	3,352	-8.1%
2	0	3	17,551	88.4%	1,459	6.9%	2,286	-0.9%
2	0	2	15,419	72.9%	1,217	2.7%	2,620	2.9%
-2	0	3	13,834	-86.8%	1,077	-2.3%	1,686	-12.6%
0	1	1	12,435	11.9%	1,442	64.8%	2,006	65.9%
-2	0	2	11,799	-68.0%	882	3.9%	1,927	-15.7%
0	-1	1	11,717	5.1%	1,260	-54.8%	2,177	-44.7%
3	0	3	10,819	97.2%	678	0.3%	1,859	1.2%
-3	0	3	7,569	-94.2%	469	7.0%	1,184	-6.3%
0	1	2	7,480	8.5%	851	57.0%	1,157	25.7%
0	0	4	7,024	-0.6%	721	5.7%	535	-10.7%
0	-1	2	6,853	0.3%	791	-52.5%	1,160	-37.4%

5.5 PLAYER VALUATIONS

[NOTE: Player Impact vs. Points has Linear Trend, so our impact score makes sense]

[TODO: Player Impact vs. Points plot]

[NOTE: Player Impact vs. Salary has semi-Linear Trend, many outliers]

[TODO: Player Impact vs. Salary plot]

[TODO: Top-20 Player Impact Table w.r.t. Probability Next Goal]

[TODO: Top-20 Player Impact Table w.r.t. Probability Next Penalty]

6 APPLICATIONS

7 CONCLUSION

It would be interesting to see the effect of shift changes, that is, when a player enters or leaves the ice, within action event sequences. As such, finding an efficient way to make use the available temporal data alongside the Markov Game Model would be an interesting study, and could be used to form simulations of NHL games. Goalies do not participate in many recorded events, it is difficult to evaluate goalies with this model, and extension methods for evaluating goaltenders are left as future work. Learning what context features perform better than others could also be an interesting study. For example, one approach is to use only the shot label, while another approach is to use shot labels augmented with the shot location and shot type as in [Krzywicki, 2005].

References

Berliner, H. (1979). The b*- tree search algorithm: A best-first proof procedure. *Artificial Intelligence*, 12(1):23–40.

Churchill, D. and Buro, M. (2013). Portfolio greedy search and simulation for large-scale combat in starcraft. In *IEEE Conference on Computational Intelligence in Games (CIG)*, pages 1–8.

Gramacy, R., Jensen, S., and Taddy, M. (2013). Estimating player contribution in hockey with regularized logistic regression. *Journal of Quantitative Analysis in Sports*, 9:97–111.

Krzywicki, K. (2005). Shot quality model: A logistic regression approach to assessing nhl shots on goal.

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the eleventh international conference on machine learning*, volume 157, pages 157–163.

Lock, D. and Schuckers, M. (2009). Beyond +/-: A rating system to compare nhl players. Presentation at joint statistical meetings.

Macdonald, B. (2011). A regression-based adjusted plus-minus statistic for nhl players. *Journal of Quantitative Analysis in Sports*, 7(3):29.

Macdonald, B. (2012). An expected goals model for evaluating nhl teams and players. In *MIT Sloan Sports Analytics Conference*.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York.

National Hockey League (2014). National hockey league official rules 2014-2015.

Rosenbaum, D. T. (2004). Measuring how nba players help their teams win.

Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Salunke, S. S. (2014). *Selenium Webdriver in Java: Learn With Examples*. CreateSpace Independent Publishing Platform, USA, 1st edition.

Schuckers, M. and Curro, J. (2013). Total hockey rating (thor): A comprehensive statistical rating of national hockey league forwards and defensemen based upon all on-ice events. In *7th Annual MIT Sloan Sports Analytics Conference*.

Schuckers, M. E., Lock, D. F., Wells, C., Knickerbocker, C. J., and Lock, R. H. (2011). National hockey league skater ratings based upon all on-ice events: An adjusted minus/plus probability (ampp) approach. Unpublished manuscript.

Schwartz, A. (1993). A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the tenth international conference on machine learning*, volume 298, pages 298–305.

Sidhu, G. and Caffo, B. (2014). Moneybarl: Exploiting pitcher decision-making using reinforcement learning. *The Annals of Applied Statistics*, 8(2):926–955.

Spagnola, N. (2013). The complete plus-minus: A case study of the columbus blue jackets. Master's thesis, University of South Carolina.

Thomas, A., Ventura, S., Jensen, S., and Ma, S. (2013). Competing process hazard function models for player ratings in ice hockey. *The Annals of Applied Statistics*, 7(3):1497–1524.

Tulsky, E., Detweiler, G., Spencer, R., and Sznajder, C. (2013). Using zone entry data to separate offensive, neutral, and defensive zone performance. In *7th Annual MIT Sloan Sports Analytics Conference*.

Weissbock, J. and Inkpen, D. (2014). Combining textual pre-game reports and statistical data for predicting success in the national hockey league. In *Advances in Artificial Intelligence*, pages 251–262. Springer.