

---

# Contents

---

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Contributions of This Thesis . . . . .	2
1.3	Thesis Structure . . . . .	4
<b>2</b>	<b>Background and Related Work</b>	<b>5</b>
2.1	The Game of Go . . . . .	5
2.1.1	Rules and Terms . . . . .	5
2.1.2	Complexity . . . . .	11
2.2	Game Tree Search . . . . .	12
2.2.1	Terminology . . . . .	13
2.2.2	Traditional Game Tree Search Algorithms . . . . .	14
2.3	Computer Go . . . . .	16
2.3.1	History . . . . .	16
2.3.2	Computer Go as Fertile Ground for Monte Carlo Tree Search	17
<b>3</b>	<b>Monte-Carlo Tree Search</b>	<b>19</b>
3.1	Terminology . . . . .	19
3.2	Basic Algorithmic Framework . . . . .	21
3.2.1	Bandit Based In-Tree Policy $\pi_{UCT}$ . . . . .	23
3.2.2	First Play Policy . . . . .	26
3.2.3	Handling of Transpositions . . . . .	27
3.3	Enhancements to Basic MCTS . . . . .	28
3.3.1	RAVE: Rapid Action Value Estimate . . . . .	28
3.3.2	PB: Progressive Bias . . . . .	30
3.3.3	PW: Progressive Widening . . . . .	31
3.4	Payout Policies $\pi_p$ . . . . .	31
3.4.1	Random Policies . . . . .	32
3.4.2	Handcrafted Policies . . . . .	33
3.4.3	Machine Learning Based Policy Design . . . . .	33

<b>4</b>	<b>Parallel Monte-Carlo Tree Search</b>	<b>35</b>
4.1	Related Work . . . . .	36
4.1.1	Parallelization for Shared and Distributed Memory . . . . .	37
4.1.2	Parallelization for Accelerator Hardware . . . . .	40
4.1.3	General Techniques for Scalability Improvements . . . . .	41
4.2	Proposed Parallelization: Distributed-Tree-Parallelization . . . . .	42
4.2.1	Distributed Simulations . . . . .	43
4.2.2	Tree Node Duplication and Synchronization . . . . .	44
4.2.3	Distributed Transposition Table . . . . .	46
4.2.4	Implementation Details . . . . .	49
4.2.5	Load Balancing . . . . .	51
4.2.6	Broadcast Nodes . . . . .	53
4.3	Experimental Results . . . . .	56
4.3.1	Setup . . . . .	56
4.3.2	Performance and Scalability . . . . .	57
4.3.3	Overhead Distributions . . . . .	68
4.3.4	Effect of Parameters . . . . .	71
4.3.5	Discussion of the Comparison of MCTS Parallelizations . . . . .	73
4.4	Chapter Conclusion . . . . .	74
<b>5</b>	<b>Move Prediction in Computer Go</b>	<b>77</b>
5.1	Background . . . . .	78
5.1.1	Terminology . . . . .	78
5.1.2	Probability Models for Paired Comparison . . . . .	79
5.1.3	Bayes Theorem . . . . .	80
5.1.4	State-Action Abstraction with Patterns in Computer Go . . . . .	81
5.2	Bayesian Move Prediction Systems . . . . .	83
5.2.1	Minorization-Maximization . . . . .	85
5.2.2	Bayesian Ranking Model . . . . .	86
5.2.3	Bayesian Approximation Ranking . . . . .	88
5.2.4	Probabilistic Ranking . . . . .	89
5.3	Experimental Results . . . . .	91
5.3.1	Setup . . . . .	91
5.3.2	Results . . . . .	91
5.4	Chapter Conclusion . . . . .	97
<b>6</b>	<b>MCTS Driven Position Analysis</b>	<b>99</b>
6.1	Motivation . . . . .	99
6.2	Background and Related Work . . . . .	100
6.3	MC-Criticality Based Semeai Detection . . . . .	104
6.3.1	Notations . . . . .	104
6.3.2	Clustering . . . . .	105

6.3.3	Player-wise, Intersection-wise and Cluster-wise MC-Criticality	109
6.3.4	Detecting and Identifying Local Fights . . . . .	112
6.3.5	Experiments . . . . .	113
6.4	Chapter Conclusion . . . . .	118
<b>7</b>	<b>Summary and Outlook</b>	<b>119</b>
7.1	Contributions . . . . .	119
7.2	Conclusions and Lessons Learned . . . . .	120
7.3	Future Directions . . . . .	122
<b>Bibliography</b>		<b>125</b>