An Evaluation Function for the Game of Amazons

Jens Lieberum

Mathematisches Institut, Univ. Basel, Rheinsprung 21, CH-4051 Basel

Abstract

Amazons is a fascinating game that shares properties of chess and go. We have written a computer program that plays Amazons. This paper reveals the secret of this program: its evaluation function. We give explicit formulas for the ingredients of our evaluation function, describe the ideas and goals behind these formulas, and discuss possible refinements. By analyzing a tournament game of Amazong against the former computer world champion 8QP we illustrate how the new features of our evaluation function can lead to victory.

Introduction

Amazons is a many-sided game - just like the game set that is typically used to play Amazons: a draughts board of size 10×10, four white and four black chess Queens (called amazons), and a supply of go pieces of one color (called arrows). The starting position and a first move of white are shown in Figure 1 on the next page. A move consists of two steps: First the player chooses an amazon of his color and moves it like a chess Queen diagonally, vertically, or horizontally as far as he likes and no obstacle (another amazon or an arrow) blocks the way. Then this amazon has to throw an arrow. Arrows also move like chess Queens. They stay at their destination square for the rest of the game and are represented by black squares in the figures of this paper. The players move alternately until one player can no longer move. This happens after at most 92 moves. The player who made the last move wins the game.

White's advantage of making the first move can be compensated by allowing black to pass n times (e.g. $n = 4$).

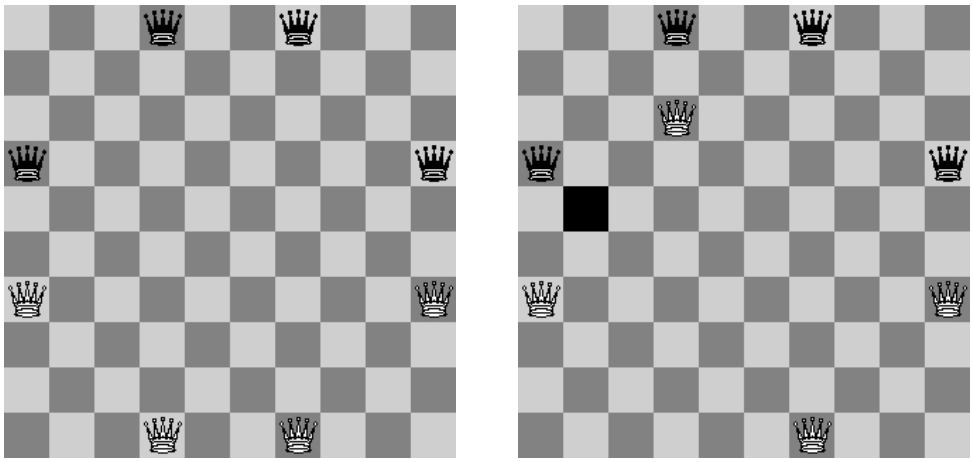

Figure 1: One good first move out of 2176 possible ones

We first heard about amazons at a workshop on combinatorial game theory at MSRI in July 2000. We were fascinated by the deepness and subtlety of 'simple' positions in Amazons that have been analyzed by E. Berlekamp ([Ber]), R. Snatzke ([Sn1], [Sn2]), M. Müller and T. Tegos ([MT]). Inspired by discussions with M. Müller about his computer program 'Arrow' and our own experiences of playing Amazons we started to write the computer program 'Amazong'. After two years of successive improvements, Amazong has won the Amazons tournament at the seventh Computer Olympiad in Maastricht in July 2002. The reader is invited to play against the java applet Amazong at http://www.math.unibas.ch/~lieberum/amazong/amazong.html.
We have described the general design of our program with a special focus on selective search in talks at the Universities of Jena and Edmonton ([Lie]). This paper complements these talks and concentrates on Amazong's evaluation function that causes its characteristic style of play, clearly distinguishes it from other programs, and is probably its main strength.

1 The Different Phases of an Amazons Game

Amazong distinguishes three phases of the game: the opening at the beginning of the game, the filling phase at the end of the game, and the main game that consists of everything else.

The opening in Amazons is the biggest challenge for computer programs because of the absence of opening theory, a branching factor of more than 1000, many situations with more than 20 reasonable moves, and the need for calculating deep variations. Human play is still superior to computers in the opening. At the computer olympiad in Maastricht in 2002 Amazong made a random choice of the first move out of three possibilities and then started to play according to the results of a selective 5- or 6-ply search. Meanwhile, the opening book has grown to a machine generated database containing more than 30000 moves following ideas of T. R. Lincke ([Lin]). However, the benefit of opening books is limited in Amazons because of the enormous complexity of this game.

The filling phase consists of those positions where each empty square on the board can be reached by at most one player by some sequence of moves. In most games this happens after approximately 50 moves. The filling phase includes positions with completely decomposed boards, meaning that amazons of different colors are separated by arrows. Then the outcome of the game can be determined by counting the number of moves left to each player. Although this problem is NP-hard ([Bur]), it is not difficult to play correctly in most positions that show up in real games on a board of size 10×10. Typically, the players stop to play and agree on the outcome of the game when the filling phase starts.

Two examples of positions from the filling phase are illustrated in Figure 2. In the position on the left side of Figure 2 it seems that white has access to two empty squares, but he has to cut off one of the empty squares with his next move. Therefore this shape is called defective territory. The position on the right side of Figure 2 is called Zugzwang because it seems that black has access to three empty squares, but if he has to move before white does, then he can only use two of the three empty squares. Amazong already tries to evaluate defective territory and many Zugzwang situations correctly before the filling phase starts. However, these parts of Amazong's evaluation function are still far from being perfect. They will not be discussed here.

Since in most Amazons games the opening book covers only the first few

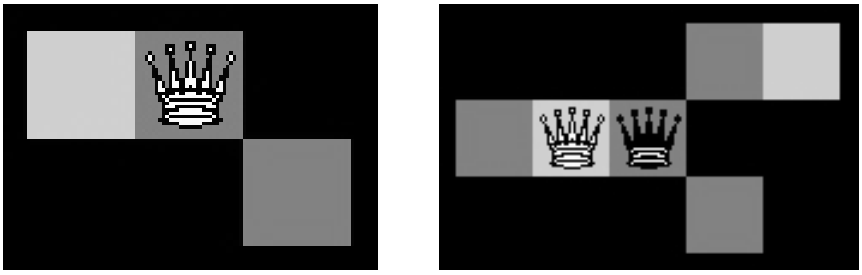

Figure 2: Defective territory and Zugzwang

moves, one has to deal with many different situations in the main game until the filling phase begins and the outcome of the game becomes clear. One possible parameter which could help to choose an appropriate strategy in each situation is the number of moves played so far. Amazong uses a different parameter to choose its strategy. This will be discussed in the next section.

2 Territorial and Positional Evaluation

The goal of the game Amazons is to have access to more empty squares in the filling phase than the other player. When player j ($j \in \{1, 2\}$, player 1 is white) has exclusive access to a region of n squares, we count these squares as n secure points of territory of player j. When both players can reach a square by some sequence of moves, it is more complicated to guess which player will eventually shoot at that square. For this purpose Amazong uses heuristics based on the following ways to measure distances on an Amazons board.

Define the distance $d_1(a, b)$ of two squares a and b as the minimal number of chess Queen moves needed to go from a to b. When there is no path, let $d_1(a, b) = \infty$. Similarly, define the distance $d_2(a, b)$ as the minimal number of chess King moves needed to go from a to b. Obviously, we have $d_1(a, b) \leq d_2(a, b)$. The distances of player j from square a are then given by

$$D_i^j(a) = \min\{d_i(a, b) \,|\, \text{the square } b \text{ is occupied by an Amazon of player } j\}.$$

On the left (resp. right) side of Figure 3 you find an example of $D_1^j(a)$ (resp. $D_2^j(a)$). In this figure the upper left corners of empty squares contain the values $D_i^1(a)$ and the lower right corners contain the values $D_i^2(a)$.

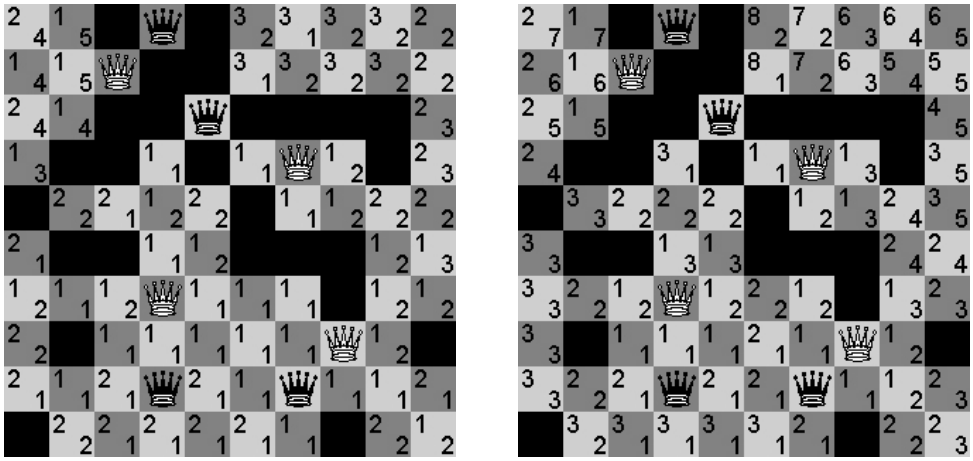

Figure 3: The minimal distances $D_i^j(a)$

All Amazons programs seem to use D_1^j in one or another way. The idea behind the definition of D_1^j is that $D_1^1(a) < D_1^2(a)$ indicates that player one has better access to the square a than player two. One heuristic for estimating the territory of player 1 is to assume that he will eventually shoot to *all* squares a with $D_1^1(a) < D_1^2(a)$. This heuristic works very well shortly before and in the filling phase. A problem of D_1^j at the beginning of the game is that a single amazon of player j in the center can cause low values of D_1^j on the whole board, but player j cannot move the amazon into all directions at once. Here D_2^j comes in. One advantage of D_2^j is its locality: often amazons have to fulfill a certain task at their position like guarding the territory in their neighborhood. Then a large value of $D_2^j(a)$ indicates that player j cannot move towards the square a without causing a positional damage, despite a possibly low value of $D_1^j(a)$. Another advantage of D_2^j over D_1^j is that it behaves more stable when the other player moves and shoots, especially when there are just a few arrows. This makes D_2^j useful for long term estimates and will stabilize the evaluation function in the beginning of the game.

We use D_i^j to assign local evaluations between -1 and 1 to each empty square. Positive values indicate an advantage of player 1. Then we sum these numbers over all empty squares in order to transform the local evaluations into global ones. One possible formula for global evaluations t_1, t_2 is given by

$$t_i = \sum_{\text{empty squares } a} \Delta(D_i^1(a), D_i^2(a)),$$

where

$$\Delta(n,m) = \begin{cases} 0 & \text{if } n = m = \infty \\ \kappa & \text{if } n = m < \infty, \\ 1 & \text{if } n < m, \\ -1 & \text{if } n > m, \end{cases}$$

and $-1 < \kappa < 1$ is a constant with $(-1)^j \kappa \leq 0$ when it is player j's turn. The number $|\kappa|$ estimates the advantage of moving first when the distances of both players to an accessible square agree. We have made good experiences with $|\kappa| \leq 1/5$, but some fine-tuning is necessary after each modification of the evaluation function. We optimize the choice of κ in order to obtain a low volatility of the evaluations during iterative deepening. This should help to avoid odd-even effects and supports aspiration search with narrow α-β-windows.

A program that uses the territorial evaluation t_1 as its evaluation function plays already quite reasonably, especially shortly before the filling phase. In contrast to that the value t_2 is useful in the beginning of the game but becomes less significant as the game goes on. The evaluations t_i share the drawback that they do not take into account that large values of $D_i^2(a) - D_i^1(a)$ are better for player 1 than small values. Therefore, other local evaluations than Δ seem to be important, too. The generic approach is to replace $\Delta(n,m)$ by some array of parameters and then to optimize these parameters. We have made good experiences with the choices

$$\begin{aligned} c_1 &= 2 \sum_{\text{empty squares } a} 2^{-D_1^1(a)} - 2^{-D_1^2(a)}, \\ c_2 &= \sum_{\text{empty squares } a} \min(1, \max(-1, (D_2^2(a) - D_2^1(a))/6)). \end{aligned}$$

Notice that in c_1 the local advantage $(D_1^1(a), D_1^2(a)) = (1, 2)$ is rewarded by 0.5 points for player 1, $(2, 3)$ by 0.25 points, $(1, 3)$ by 0.75 points, and squares a with $(D_1^1(a), D_1^2(a)) = (n, n)$ contribute 0 points. Other tuples are of minor practical importance for c_1. In contrast to c_1, c_2 depends only on $D_2^2(a) - D_2^1(a)$ and only large differences indicate a clear advantage of one player.

Now we have to combine the values t_i and c_i into one evaluation function. A weighted sum with static weights does not seem to be appropriate for this because the importance of the values t_i and c_i varies during the game. Therefore we should first try to compute the expected number W of moves needed until the filling phase starts. Instead of trying to estimate W directly we simply define

$$w = \sum_a 2^{-|D_1^1(a) - D_1^2(a)|},$$

where we sum over all empty squares a with $D_1^1(a) < \infty$ and $D_1^2(a) < \infty$. Obviously, we have $w = 0$ if and only if the position belongs to the filling phase and typically w decreases with the number of moves played. Therefore we expect that a good estimate of W will be some function of w. For our purposes, w is just as good as W. Now define an evaluation t as

$$t = f_1(w)t_1 + f_2(w)c_1 + f_3(w)c_2 + f_4(w)t_2,$$

where $(f_i)_i$ is a partition of 1 (meaning $0 \leq f_i(w)$ and $\sum_i f_i(w) = 1$). The exact form of the functions f_i is a problem of parameter optimization. Our choice of f_1 has been guided by the observation that t_1 becomes more and more important during the main game and gives very good estimates of the expected territory shortly before the filling phase. Therefore f_1 is monotonously decreasing and satisfies $f_1(0) = 1$. The counterpart of t_1 is t_2. It rewards balanced distributions of the own amazons on the board or helps to hinder the other player from reaching such a distribution. This is most important at the beginning of the game. The values c_1 and c_2 allow to detect finer properties of the position than t_1 and t_2 alone, because they depend on the quality of local advantages. They support good positional play in the opening and a smooth transition between the beginning and later phases of the game. This is most evident for t_1 and c_1: while at the end of the game only t_1 counts, c_1 rewards moves in earlier phases of the game that replace

clear local disadvantages by small disadvantages and small advantages by clear advantages.

3 Mobility of Individual Amazons

Amazong is trying to enclose amazons of the other player inside of small regions at the beginning of the game. Compared to other computer programs, this is Amazong's main strength. In this section we will present a modification of the evaluation function t (see section 2) that is responsible for this behavior.
Enclosing amazons typically does not cause an appropriate change of t (and especially of t_1) in the beginning of the game. This can be explained as follows: when a single amazon A of player 1 is enclosed in some small region of n points, then the Amazons board is divided into two parts: the inside and the outside of that region. Player 1 has exclusive access to the territory on the inside. This contributes n points to t. On the outside, some active amazons of player 1 might overshadow the missing influence of A in D_1^1. In addition, some amazons of player 2 that have helped to enclose A might not be in optimal positions but often have a large potential to improve their positions. The problem that A cannot reach the outside for the rest of the game is not reflected in the computation of t. The disadvantage of the enclosed amazon often starts to affect t several moves later. Then it is too late. Therefore a correction term m is needed to take into account the mobility of individual amazons. Since active amazons can overshadow bad positions of passive amazons in the evaluation function t it seems more important to punish passive and enclosed amazons than to support active amazons in this correction term. To compute m quickly, consider first the number $N(a)$ of empty squares that can be reached from a by a single move of a chess King. The numbers $N(a)$ can be updated incrementally during the search inside of functions doMove and undoMove. For an amazon A of player j on the square a, let

$$\alpha_A = \sum_b 2^{-d_2(a,b)} N(b),$$

where we sum over all squares b with $d_1(a,b) \leq 1$ and $D_1^{3-j}(b) < \infty$. When $\alpha_A = 0$ we say that the amazon A is enclosed. Examples of the

values $N(a)$, α_A and of enclosed amazons are shown in Figure 4. For example, for the white amazon A in the upper left corner of the figure on the left, we compute $\alpha_A = 7 + 6 + 5 + 3 + 3 + (5 + 4 + 7 + 4)/2 + 5/4 = 35.25$. The two white amazons in the lower right corner in this figure are enclosed.

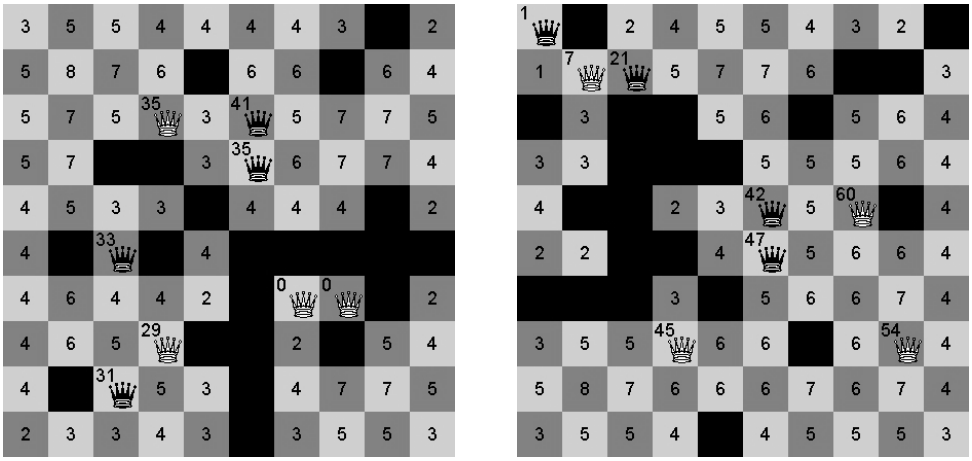

Figure 4: Neighbors $N(a)$ of empty squares a and the values α_A

We have learned in discussions with experienced amazons players that at the beginning of a game on a board of size 10×10 enclosed amazons should be punished by a malus of at least 10 points. In general, we use w from the last section to define

$$m = \sum_{\substack{\text{amazons } B \\ \text{of player 2}}} f(w, \alpha_B) - \sum_{\substack{\text{amazons } A \\ \text{of player 1}}} f(w, \alpha_A)$$

for a suitable function $f \geq 0$. The exact choice of f is the hardest optimization problem in our evaluation function $t + m$, so we restrict our description to the properties of f that did not change during our experiments: f satisfies $f(0, y) = 0$ and $\frac{\partial f}{\partial x}(x, y) \geq 0$ because the longer an amazon is enclosed before the filling phase starts the bigger is the disadvantage. Furthermore, f satisfies $\frac{\partial f}{\partial y}(x, y) \leq 0$ because a low value of α_A corresponds to a passive position of the amazon A. The last dependence is not linear. We have made good experiences with functions f that satisfy $2f(w, 5) < f(w, 0)$. This can

be explained as follows: $\alpha_A \approx 5$ indicates that the amazon A is almost enclosed. However, there is a big difference between an enclosed and an almost enclosed amazon. The other player possibly has to move an own amazon B to an unfavorable square to prevent A from escaping. The resulting change of t then has to be compensated for by m. In addition, the task of guarding A makes the amazon B less mobile.
The big difference between enclosed and almost enclosed amazons can be seen on the right side of Figure 4: white can enclose the black amazon B with $\alpha_B = 1$ in his next move, but then black can reply by enclosing the white amazon, too. Similarly, the task of guarding the white amazon in the upper left corner puts the black amazon B with $\alpha_B = 21$ in danger of getting enclosed.

4 Comparison between $\mathbf{t_1}$ and $\mathbf{t + m}$

In this section we compare our evaluation function $t+m$ with t_1 by using the game Amazong vs. 8QP played at the 7th Computer Olympiad in Maastricht. The position after 26 moves in this game is shown in Figure 3 on page 5. Amazong won the game by 8 points, mainly due to the enclosed black amazon in the upper left corner. Figure 5 shows how t_1 and the different components of $t + m$ varied during the game.
In this diagram the values t_i are computed using $|\kappa| = 0.1$. The lines corresponding to t_1, $t + m$ and m should be clearly visible in the diagram. In move 13 white enclosed the black amazon which causes the maximum of the dashed line corresponding to m. Notice that at this point the evaluation $t + m$ predicts the outcome of the game very well and differs from t_1 by more than 18 points. Then $t + m$ and t_1 become more and more related and finally coincide when the filling phase is reached.
As expected, the values c_2 and t_2 are more stable than c_1 and t_1. In addition, c_2 and t_2 are positive in almost all positions of the game. This indicates that the evaluation function of $8QP$ does not consider King move distances. Therefore $8QP$ puts up no resistance against Amazong maximizing these components of $t + m$.

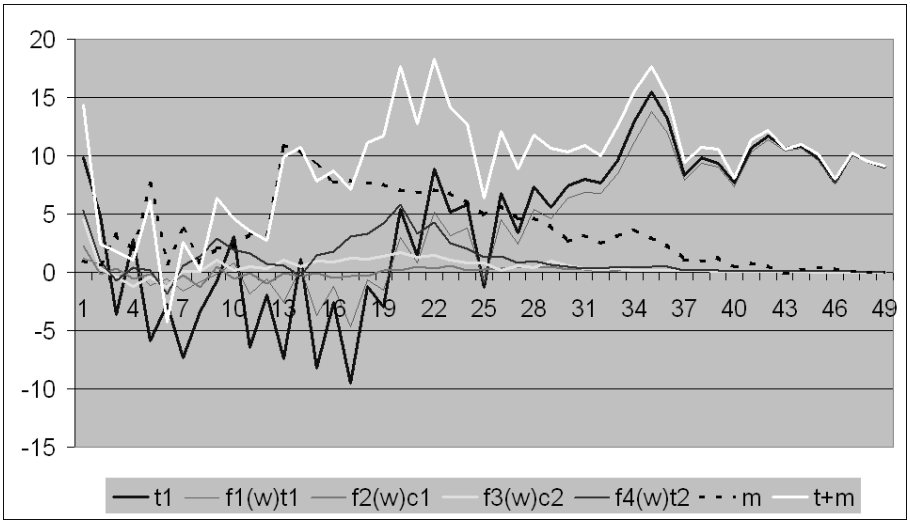

Figure 5: The components of the evaluation function $t + m$ during a game

5 Refinements and Outlook

Consider positions with regions that are (almost) separated by arrows. How much is it worth when one player has a majority of amazons inside of such a region? Instead of looking for a general answer to this difficult question, we simply observe that the territorial evaluation t has the tendency to underestimate the advantage of the majority. A possible correction term of t could take into account the distances between each empty square and each amazon. However, the computations of these values would take almost four times longer than the computations of $D_i^j(a)$. Therefore, it seems more appropriate to compute only the numbers of amazons A_ν of player j on squares b_ν that satisfy $d_i(a, b_\nu) = D_i^j(a)$. These numbers can be computed efficiently together with $D_i^j(a)$. They are useful as additional inputs of refined definitions of c_i and t_i. In addition to these corrections, the disadvantage of having a majority of amazons in a *small* region early in the game should be reflected by m. This situation is not treated correctly by m because when amazons of both players are inside of one region the involved amazons are not considered as being enclosed.

Another refinement concerns the distribution of amazons on the board. In the opening it is desirable (especially for black) to reach a position with

exactly one amazon in each corner of the board. The distances from such a distribution can be used to improve the evaluation function in the opening phase.
In other experiments, we weighted squares in the computations of c_i and t_i. The weights depended on w and the distance of the square from the center of the board. It is difficult to judge the importance of this variation.
Another idea for improvements is to repeat the constructions of section 2 for other distance functions like $d_1 + d_2$ or $2d_1 + d_2$ (or estimates of these distances that can be computed more efficiently). One has to decide very carefully how many different distance functions one should use, because each additional distance function slows down evaluations considerably.
The biggest weakness of our evaluation function seems to be the underestimation of large territorial frameworks at the beginning of the game. Our hope is to incorporate ideas from ([Lor]) to overcome this weakness. This is difficult because in many situations one has to make a choice between two plans that are often incompatible: chasing and enclosing amazons or building large territorial zones. The decision which plan is the more promising one in an actual position is a challenge for the next generation of Amazons programs.

References

[Ber] E. R. Berlekamp, *Sums of* $2 \times N$ *Amazons*, Game Theory, Optimal Stopping, Probability and Statistics: Papers in honor of Thomas S. Ferguson (edited by F. T. Bruss and L. le Cam), Institute of Mathematical Statistics Lecture Notes – Monograph Ser., Vol. 35 (2000), 1–34.

[Bur] M. Buro, *Simple Amazons Endgames and Their Connection to Hamilton Circuits in cubic subgrid graphs*, Proceedings of the Second International Conference on Computers and Games, CG00, vol. 2063 (edited by T. Marshland and I. Frank), Lecture Notes in Computer Science (2001), 250–261.

[Lie] J. Lieberum, *Selective Search in Amazons*, powerpoint presentation used in talks at the Universities of Jena and Edmonton, available at http://www.math.unibas.ch/~lieberum/amazong/amazong.html.

[Lin] T. R. Lincke, *Strategies for the Automatic Construction of Opening Books*, Proceedings of the Second International Conference on Computers and Games, CG00, vol. 2063 (edited by T. Marsland and I. Frank), Lecture Notes in Computer Science (2001), 74–86.

[Lor] R. Lorentz, *Finding Territory in Amazons*, The Seventh Computer Olympiad Computer-Games Workshop Proceedings, Technical Reports in Computer Science, Universiteit Maastricht.

[MT] M. Müller and T. Tegos, *Experiments in Computer Amazons*, More Games of No Chance (edited by R. Nowakowski), 243–260.

[Sn1] G. Snatzke, *Exhaustive Search in the Game Amazons*, More Games of No Chance (edited by R. Nowakowski), 261–278.

[Sn2] G. Snatzke, *New Results in Exhaustive Search in the Game Amazons*, University of Jena preprint 2002.