

The CodeCup, an annual game programming competition.



Willem van der Vegt (willem@informaticaolympiade.nl) is a teachers trainer in mathematics and computer science at Windesheim University for Professional Education in Zwolle, the Netherlands. He is also the contest director of the Dutch Informatics Olympiad. He joined the IOI nine times.

Summary: The annual game programming competition also known as the CodeCup offers an environment for a set of programming problems that are only partial algorithmic. After an introduction of the competition and its history and development, I will point out some of the interesting topics that arose in these competitions. Of course we have a list of wishes for the future of our competition. At the end of this paper I will present some of the questions that are still open using these kinds of programming tasks.

Introduction:

Games offer an intriguing environment for developing computer programs. Computer chess is object of studies in computer science for a long time, and it offers a lot of methods to use in designing games [1]. But unlike the games that were used in the past IOI's¹, in our competition we tend to choose games that do not have a known perfect solution. This way we encourage participants to investigate new things and to compete with their own original ideas.

History:

The annual CodeCup-contest (www.codecup.nl) is a side event for the Dutch Informatics Olympiad (NIO). Contestants for the CodeCup make a program for the final task of the first round of the NIO. They have to write a program to play a game. A working program that not makes errors in competition will score 80 out of 100 for the NIO. With little effort and limited knowledge in advance it is often possible to make a working program, also because example programs with working IO are offered. The main challenge is to find good algorithms to be able to beat your opponents. Since the CodeCup started every two weeks test competitions are organized that give you an idea of the performance of your program.

In 1996 we started our first competition with a solitaire game. We prepared a number of random input sets, and after every round of 10 games the worst performing programs were discarded. In 1997 we did the first two persons board game, and we started a system with group competitions and a final group to find the winner. In 1998 we started to present the result on the Internet (alas, they are not available any more) and since 1999 a full competition was played [2]. Submissions were made by email or by sending a floppy. The organization had to do a lot of testing to be able to run the tournament.

¹ For instance: IOI'96 Day 1, Problem 1: A Game. In the problem description is stated: "If the board initially contains an even number of elements, then the first player has a winning strategy. You are to write a program that implements the strategy of the first player to win the game."

Starting in 2003, we extended this task and opened the CodeCup-contest for everyone who was interested. On the CodeCup-website the game is presented, contestants can submit their programs that are automatically tested and the tournament is run, preceded by a number of test competitions.. In 2004 the competition formula changed: We play a number of rounds in a Swiss tournament; then the best placed programs meet each other in a full competition.

All games of the past ten years are shortly described in Appendix A.

Programming issues in these contests:

For all games programs have to be written that are able to perform these tasks:

1. Representing a game and a game history.
2. Generating all possible moves.
3. Maintaining a kind of game tree to be able to investigate future moves.
4. Evaluating game positions and selecting moves.

Representing a game and a game history.

This includes topics like:

- Developing a data structure for a game board.
- Elaborating notions like neighbourhood, adjacent cells, rows and columns.
- Realizing the proper input and output procedures.

Before the years of the CodeCup, contestants had to make programs that read a game history and wrote a move. A game was played by starting a batch file that called each program in turn, providing it with the proper input and analyzing the output. You had to do your thinking all over again each move, because you had no way to save your intermediate results.

Nowadays, the contest system starts both players programs and they are temporary stopped where it is the turn for the other program to move. So the way to communicate slightly changed. You are able to reuse the results of an evaluation of all positions and to do some pre-processing at the start of the game. Alas this implied that it was not so easy to make the competition software yourself in order to test your program. The Caia-project was started in order to give contestants an environment to test his own players; it is described at the CodeCup-website. This description also gives a good idea on the technical background of our competition software.

Generating all possible moves.

In order to generate all possible moves you need to implement all the given rules. In Caissa for instance, moves were only allowed if a connection rule was still applied. The check for this connection rule is a classic algorithmic problem (are all available places on the game board still connected). In Tac-Tic Turn and in Lamistra you were not allowed to take back your last move, so keeping track of the game history is needed. In Lasca it was possible to surrender or to offer a draw, so these possibilities should occur when generating al list of all possible moves.

In some games it is not needed to be able to generate all possible moves to enter the contest. In Turn Right, this years game, generating all closed paths on a game board with a

given maximum number of edges is quite a heavy programming task. Less experienced programmers can start with submitting a program that doesn't use all of the possible moves, in order to develop a proper working program. During the test competitions these contestants are challenged to improve their submission by finding all the moves they first neglected.

Maintaining a kind of game tree to be able to investigate future moves.

This is of course the area of classical game theory, with stuff like minimax reasoning and alpha-beta-pruning [1]. The number of branches in the game trees varied from 6 (for instance in the initial game board of Dao) to 1800 (like in the initial game board in Pas), so finding a good value for the search depth was useful in one year, but tricky in other competitions. The winner of the Caissa-competition discovered a very deep combination where a piece was given away in order to win the game a few moves later. He was able to apply this combination is the match with the runner up, so it was very impressive.

Evaluating game positions and selecting moves.

The use of a good evaluation function, in order to be able to distinguish different positions in the game, is of course important. Sometimes the rules of a game (like Pas or Turn Right) imply that there are two levels to score points. It is very disappointing to gather a lot of points on the secondary level, and then still to lose the game.

Two of the games we used turned out to be solved. Tac-Tic Turn could always be won by the first player; Dao was a solid draw. Still the competitions for these games were full of surprises. The competition for Tac-Tic Turn was won by two contestants. One of these used a program that lost one of its games, when first to move. But its ability to defeat other programs when not playing a perfect game helped it to survive in the tournament. In the Dao-competition one of the contestants developed a kind of fuzzy logic system in order to be able to beat as many week opponents as possible.

Future wishes:

We would like to have a competition with a n-persons game (with $n > 2$), like a card game.

We once used the idea of offering a draw, of the possibility to surrender (for a slightly better score than losing). For some games this can be a nice option again.

We like a lot more contestants from all over the world.

Discussion:

We like the way these contests offer us the possibility to present more complex tasks with different problems within them. Scoring the results of these tasks is however not easy.

Until now, a contestant of our first round is able to gain 80 out of 100 points for submitting a working game program. The final 20 points are available for the position that his program reaches in the tournament.

When you look at the programming issues above, only the first and a part of the second issue have to be realized in order to be able to score these 80 points. In a lot of competitions there were smart contestants that were able to score these points with little effort.

The check whether a program performs all right is made in two steps: In the first test the program plays against two of the players of the organization. When an error occurs, the program is rejected. The other part of the test is the actual tournament; for all errors encountered points are subtracted.

Conclusions:

Games and tournaments like the CodeCup offer a great context for programming competitions. The CodeCup-contest system offers an easy environment for presenting a game, entering or re-entering submissions, testing programs and running a tournament.

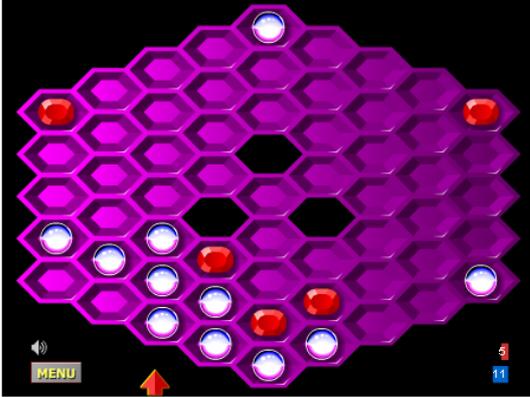
In programming a game a lot of computer science problems need to be solved. Only when these issues are dealt with it is possible to enter a competition.

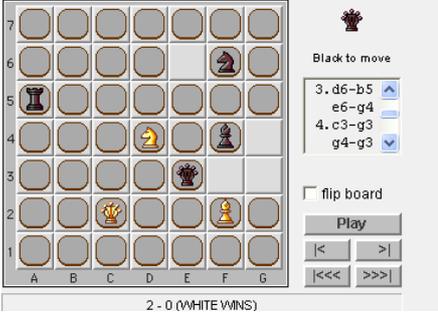
In our competition only a small part of the score is given for the algorithms that give your program the chance to win a tournament. Finding a good way for translating the results in the tournament into a score for our Olympiad is not so easy.

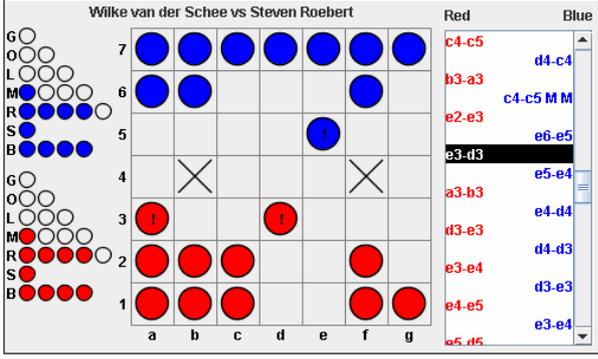
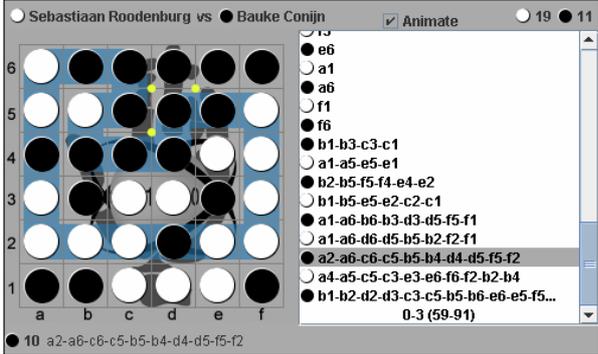
Literature:

1. Computer chess compendium, David Levy, 1988, Springer-Verlag. New York.
2. De wedstrijden binnen de Nederlandse Informatica Olympiade, Willem van der Vegt, in Tinfon 99/3, journal for computer science and education (Dutch).

Appendix A: A history of the CodeCup and its predecessor.

Year	Game	Description	Contestants
1996	Rack-O	Played as a solitaire game: You have a rack with 10 places, they are filled with numbered cards. In every move you pick a new card and you have to put in your rack, discarding one of the cards you already had. Goal is to achieve a increasing sequence of cards.	37
1997	Hexxagon	 <p>See http://hexxagon.freeonlinegames.com/</p>	37
1998	Tac-Tic Turn	 <p>Make 4 in a row by adding pieces to the field, or by turning one of the 9 2x2 squares.</p>	63
1999	PAS	 <p>Place your coloured stones so that they will fit with the others; try to place more stones than your opponent.</p>	51

2000	Lasca	 <p style="text-align: center;">THE GAME OF LASCA</p> <p>This is a checkers like game where you take the pieces you have captured with you.</p>	28																																																																
2001	Susan	 <p>Place your pieces in turn and try to enclose a piece of your opponent. See: http://www.stephen.com/sue/sue.html</p>	42																																																																
2002	DAO	 <p>This game was modified for our contest, because a lot of strategy was revealed on the website www.playdao.com</p>	20																																																																
2003	Caissa	 <p style="text-align: center;">2 - 0 (WHITE WINS)</p> <p>This game of Christian Freeling is described at http://www.mindsports.net/CompleteGames/Checkmate/Caissa.html</p>	32																																																																
2004	Lamistra	<table border="1" data-bbox="683 1603 979 1868"> <tr><td>A</td><td>B</td><td>A</td><td>L</td><td>O</td><td>N</td><td>E</td><td>13</td></tr> <tr><td>X</td><td>I</td><td>B</td><td>I</td><td>U</td><td>L</td><td>A</td><td>3x1</td></tr> <tr><td>I</td><td>Q</td><td>I</td><td>P</td><td>T</td><td>I</td><td>C</td><td>2</td></tr> <tr><td>S</td><td>N</td><td>O</td><td>B</td><td>W</td><td>A</td><td>Y</td><td>3+2</td></tr> <tr><td>E</td><td>L</td><td>S</td><td>A</td><td>O</td><td>N</td><td>M</td><td>2+1</td></tr> <tr><td>S</td><td>H</td><td>E</td><td>B</td><td>R</td><td>A</td><td>A</td><td>2+2</td></tr> <tr><td>W</td><td>E</td><td>S</td><td>Y</td><td>K</td><td>Y</td><td>D</td><td>1</td></tr> <tr><td>8</td><td>1+1</td><td>13</td><td>2+3</td><td>13</td><td>5</td><td>3</td><td>80</td></tr> </table> <p style="text-align: center;">Final position for one of the players</p> <p>This is a game where players in turn have to choose a letter and place it in a vacant cell on their own game board. The goal is to form more or longer valid words than the opponent.</p>	A	B	A	L	O	N	E	13	X	I	B	I	U	L	A	3x1	I	Q	I	P	T	I	C	2	S	N	O	B	W	A	Y	3+2	E	L	S	A	O	N	M	2+1	S	H	E	B	R	A	A	2+2	W	E	S	Y	K	Y	D	1	8	1+1	13	2+3	13	5	3	80	54
A	B	A	L	O	N	E	13																																																												
X	I	B	I	U	L	A	3x1																																																												
I	Q	I	P	T	I	C	2																																																												
S	N	O	B	W	A	Y	3+2																																																												
E	L	S	A	O	N	M	2+1																																																												
S	H	E	B	R	A	A	2+2																																																												
W	E	S	Y	K	Y	D	1																																																												
8	1+1	13	2+3	13	5	3	80																																																												

2005	Lamistra	 <p>Latent Mini Stratego, a game where you can postpone the decision of the value of your pieces until you are forced to do so.</p>	31
2006	Turn Right	 <p>Try to make 3x3 squares, all filled with your own pieces, by placing your pieces in a clever way and by choosing closed path to turn them one step in the right direction.</p>	32?