Nested Monte-Carlo Search

Tristan Cazenave
LAMSADE
Université Paris-Dauphine
Paris, France

Summary

- Nested Monte-Carlo Search
- Morpion Solitaire and Same Game
- Constraint Satisfaction Algorithms
- Kakuro and Sudoku
- Bus Regulation
- Conclusion

Iterative Sampling

- Principle :
- Play random games
- Retain the best

Iterative Sampling

```
int sample (position)
1 while not end of game
2     position = play (position, random move)
3 return score
```

Nested Monte-Carlo Search

- Principle :
- Play random games at the base level
- For each move at level l+1, play a level l search, and play the move with the best search result.
- Important : memorize the best sequence so far

Nested Monte-Carlo Search

```
int nested (position, level)
1 best score = -1
2 while not end of game
3     if level is 1
4       move = argmax_m (sample (play (position, m)))
5     else
6       move = argmax_m (nested (play (position, m), level - 1))
7     if score of move > best score
8       best score = score of move
9       best sequence = seq. after move
10    bestMove = move of best sequence
11    position = play (position,bestMove)
12 return score
```

Iterative Nested Monte-Carlo Search

- Principle :
- Play nested random games of a given level
- Retain the best

Iterative Nested Monte-Carlo Search

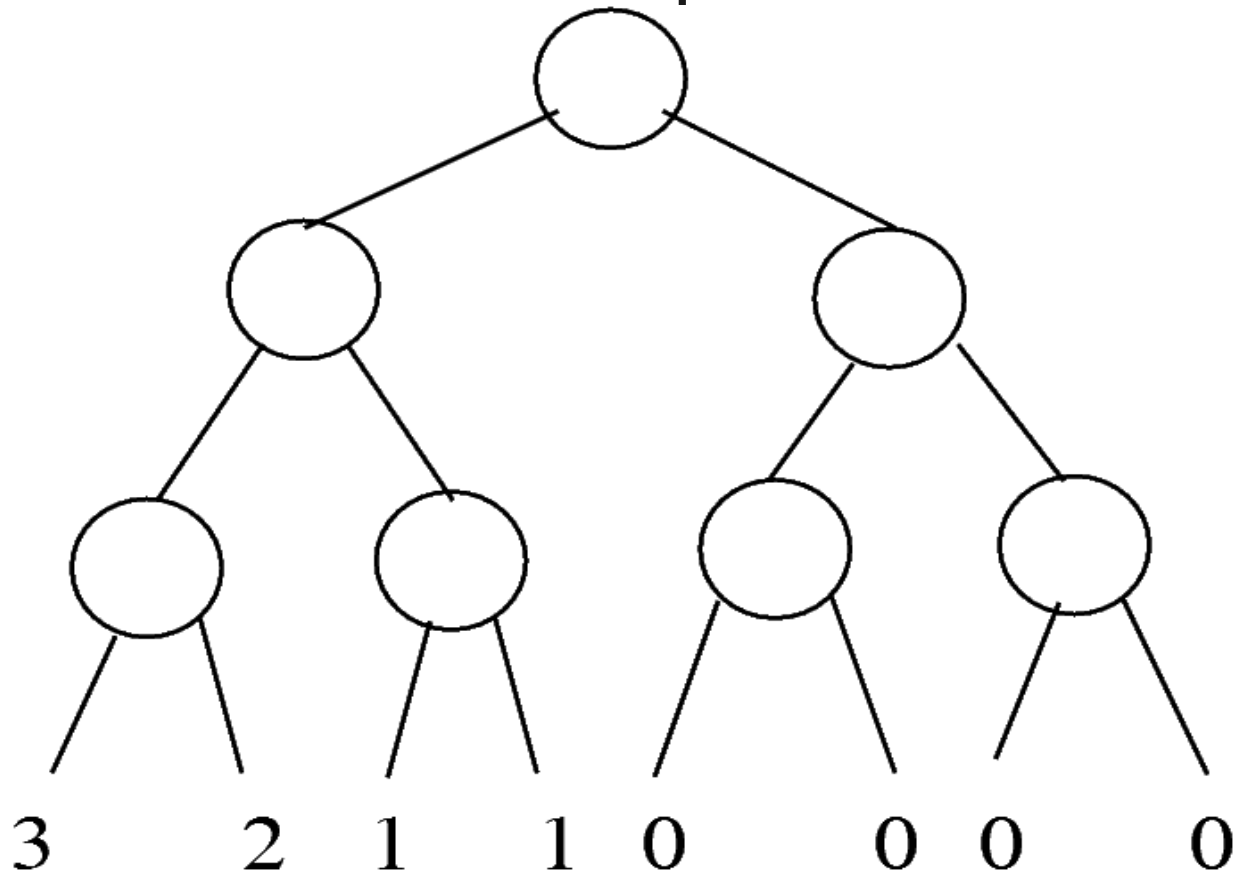
```
int iterativeNested (position, level)
1 bestScore = -1
2 while time left
3     score = nested (position, level)
4     bestScore = max (bestScore, score)
5 return bestScore
```

Analysis

- Analysis on two very simple abstract problems.
- Search tree = binary tree.
- In each state there are only two possible moves: going to the left or going to the right.

Analysis

- The scoring function of the leftmost path problem consists in counting the number of moves on the leftmost path of the tree.

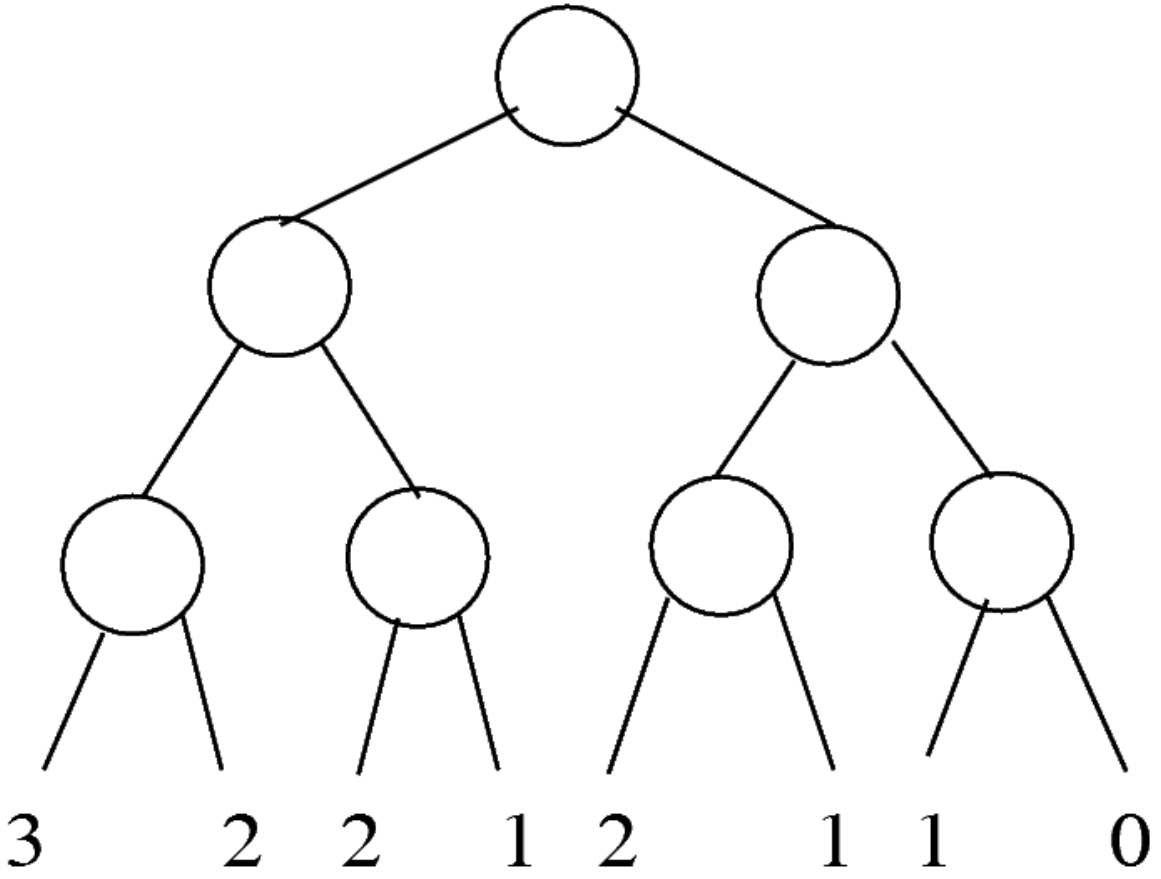


Analysis

- Sample search : probability 2^{-n} of finding the best score of a depth n problem.
- Depth-first search : one chance out of two of choosing the wrong move at the root, so the mean complexity > 2^{n-2}.
- A level 1 Nested Monte-Carlo Search will always find the best score, complexity is n(n-1).
- Nested Monte-Carlo Search is appropriate for the leftmost path problem because the scores at the leaves are extremely correlated with the structure of the search tree.

Analysis

- The scoring function of the left move problem consists in counting the number of moves on the left.

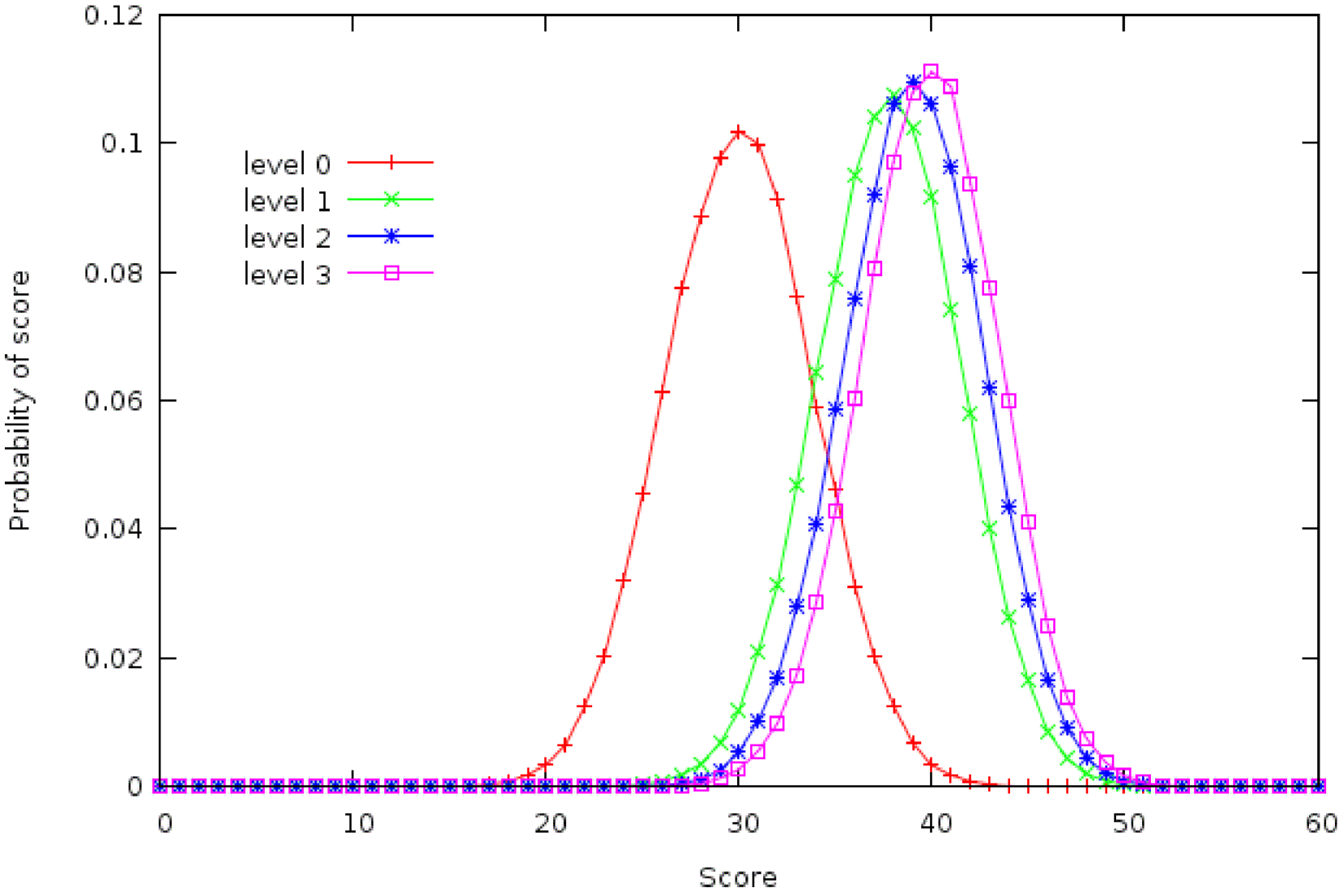


Analysis

- The probability distribution can be computed exactly with a recursive formula and dynamic programming.
- A program that plays the left move problems has also been written and results with 100,000 runs are within 1% of the exact probability distribution.

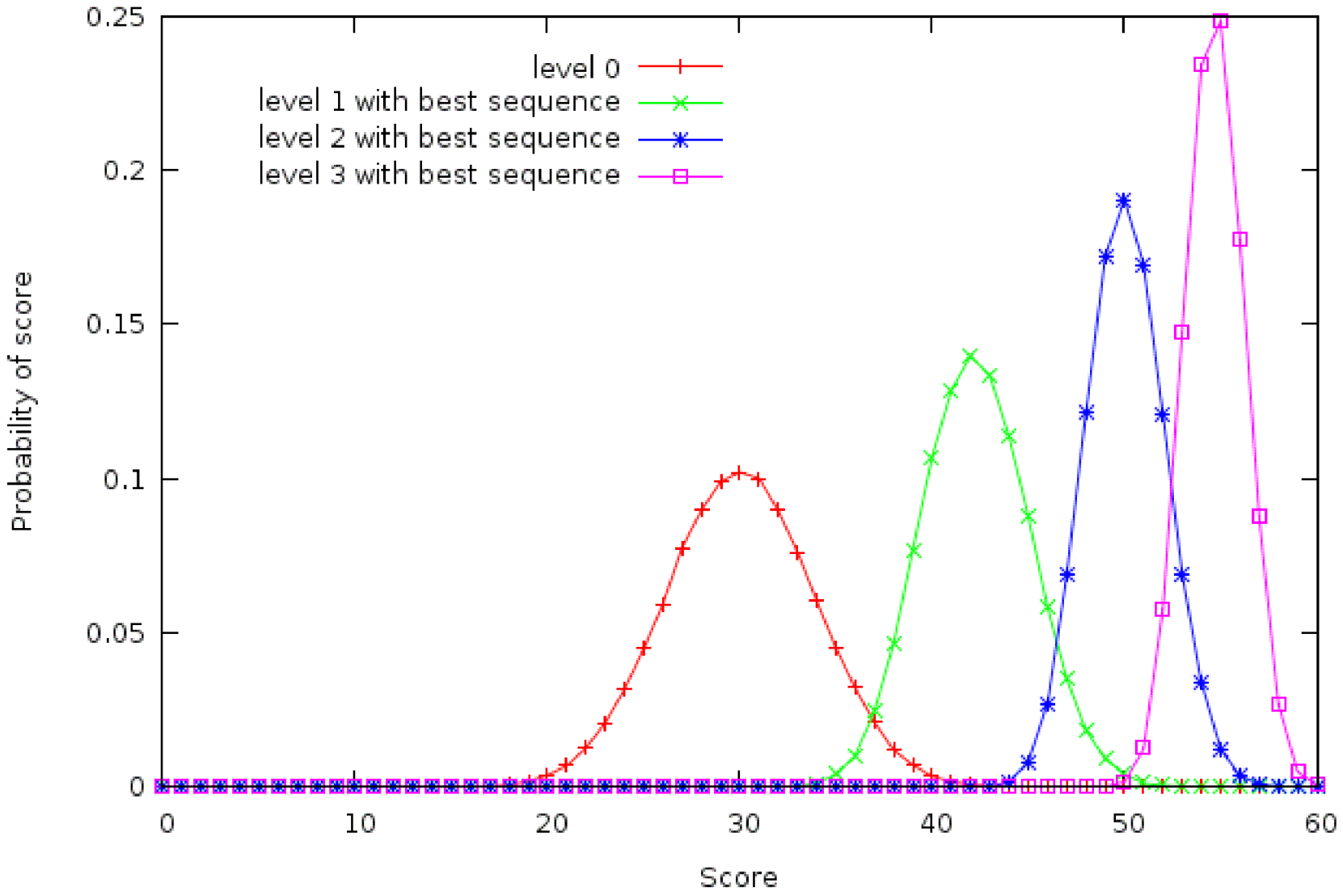
Analysis

- Distributions of the scores for a depth 60 left move problem and different search levels.



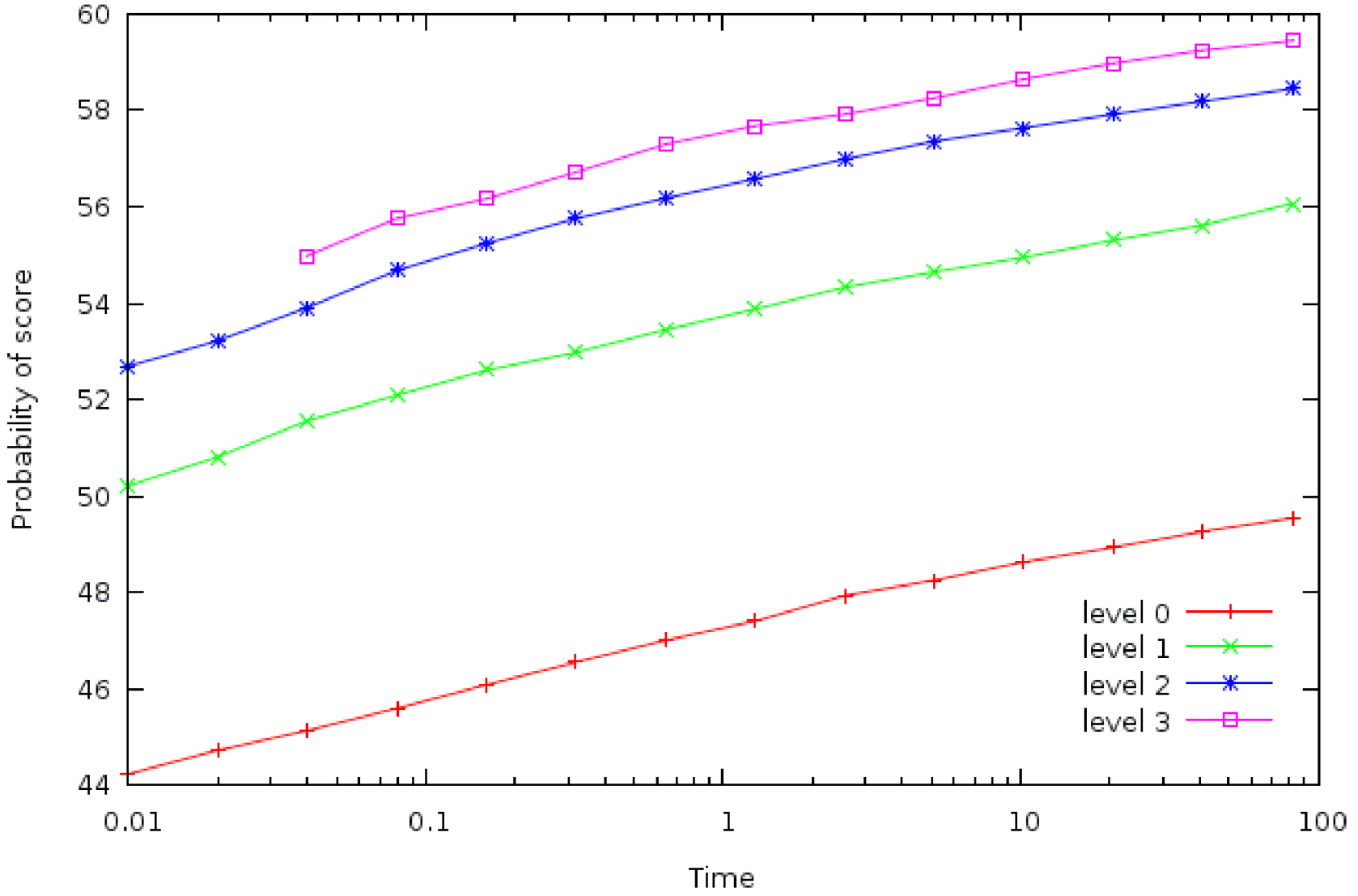
Analysis

- Distributions of the scores for a depth 60 problem and memorization of the best sequence.



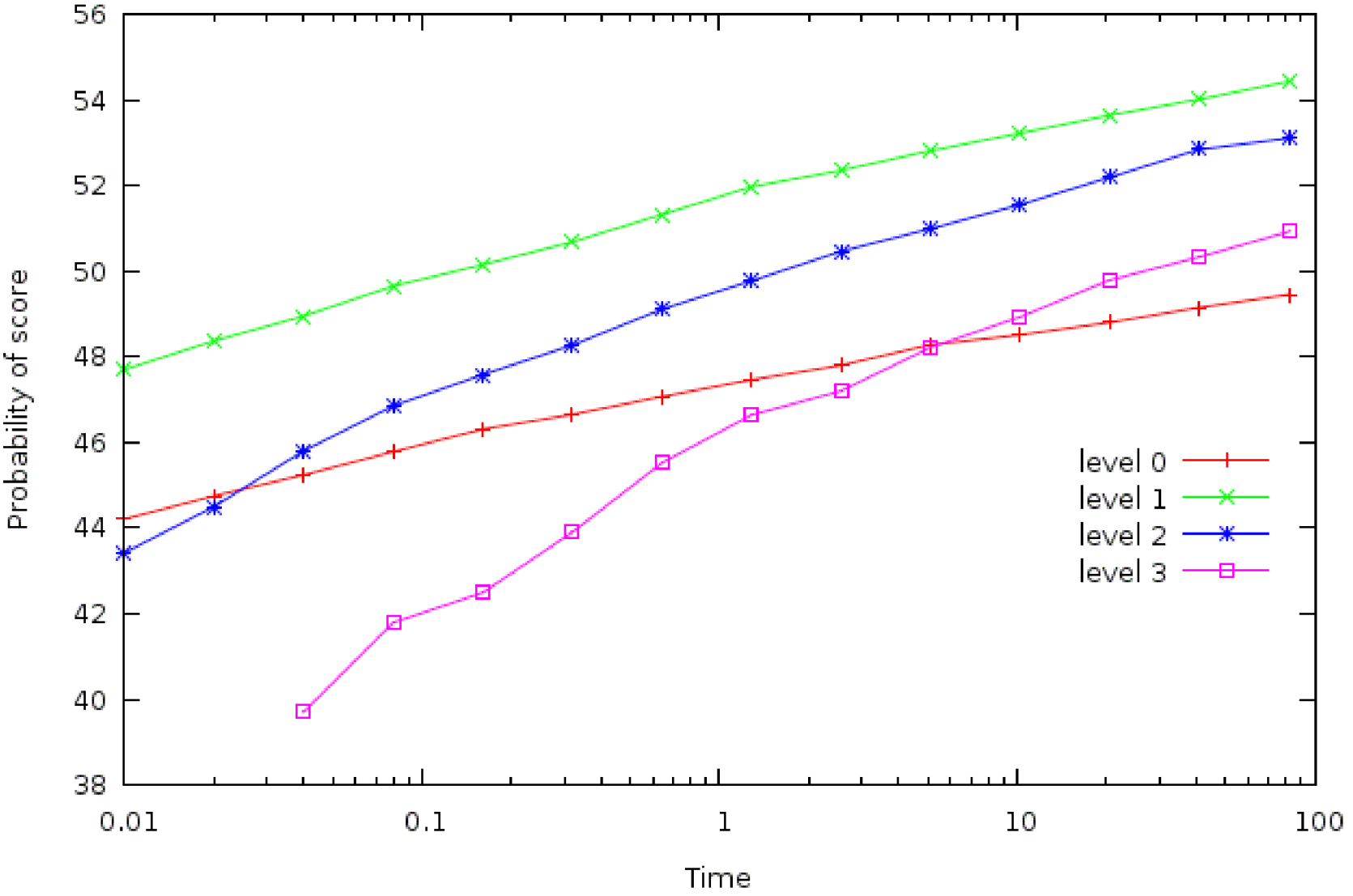
Analysis

- Mean score in real time with memorization



Analysis

- Mean score in real time without memorization



Analysis

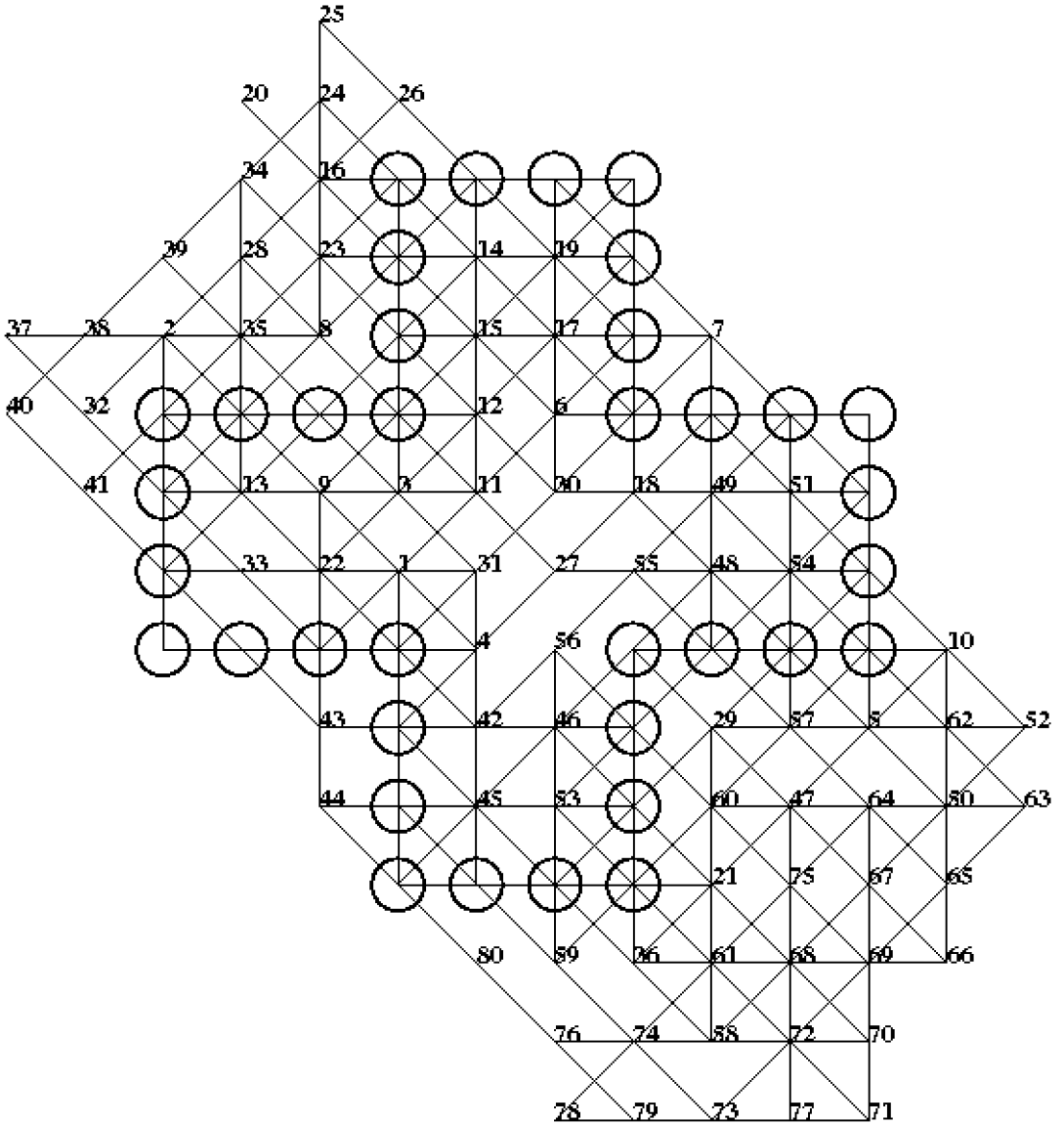
- Memorizing the best sequence is necessary for nested search to work at levels greater than one.
- Even at level one it is better.
- Nested search with memorization improves the results in a real time setting and improves the distribution of scores.

Morpion Solitaire

- Morpion Solitaire is an NP-hard puzzle and the high score is inapproximable within $n^{1\text{-epsilon}}$
- A move consists in adding a circle such that a line containing five circles can be drawn.
- In the disjoint version a circle cannot be a part of two lines that have the same direction.
- Best human score is 68 moves.
- Level 4 Search => 80 moves, after 5 hours of computation on a 64 cores cluster.

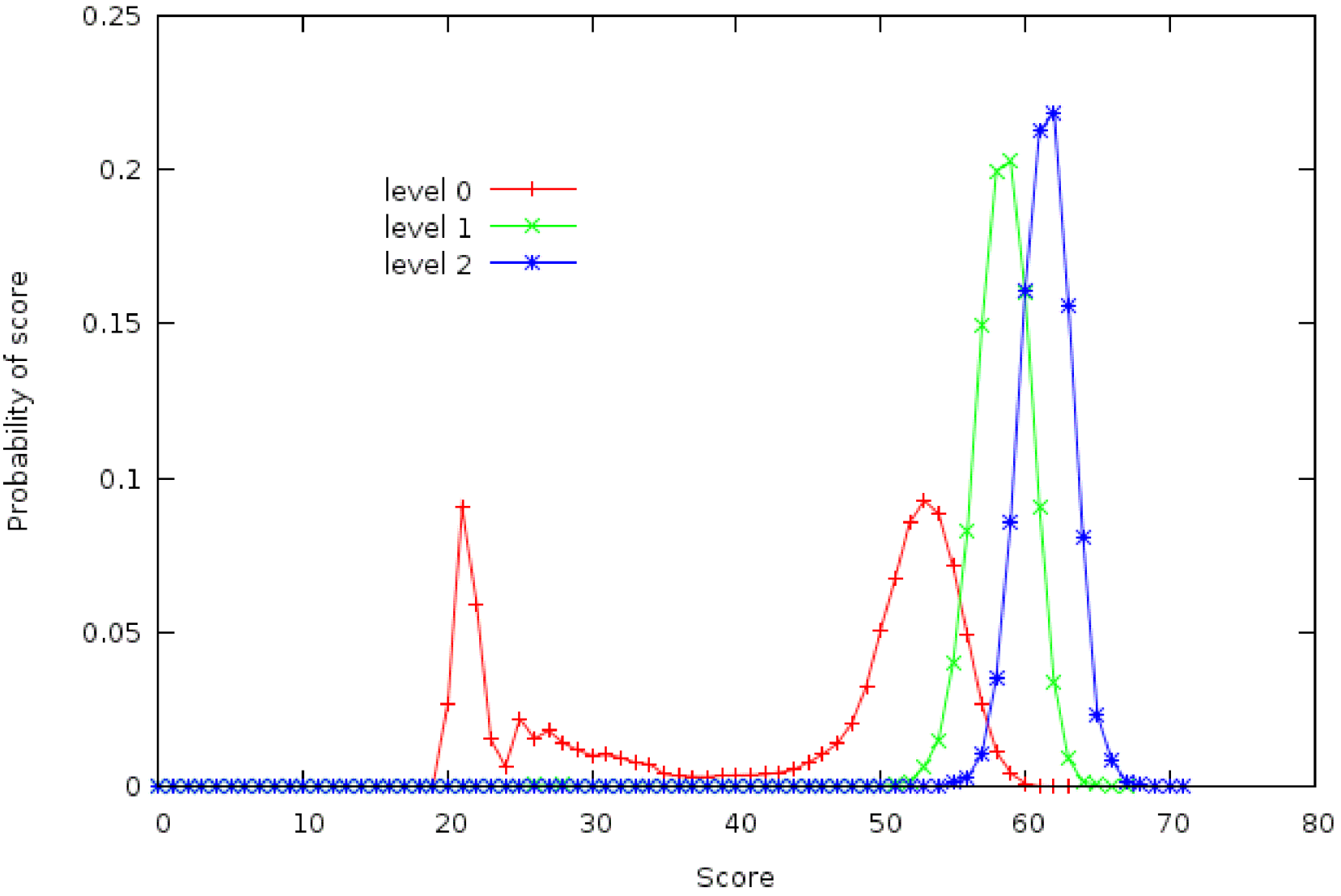
Morpion Solitaire

- World record:



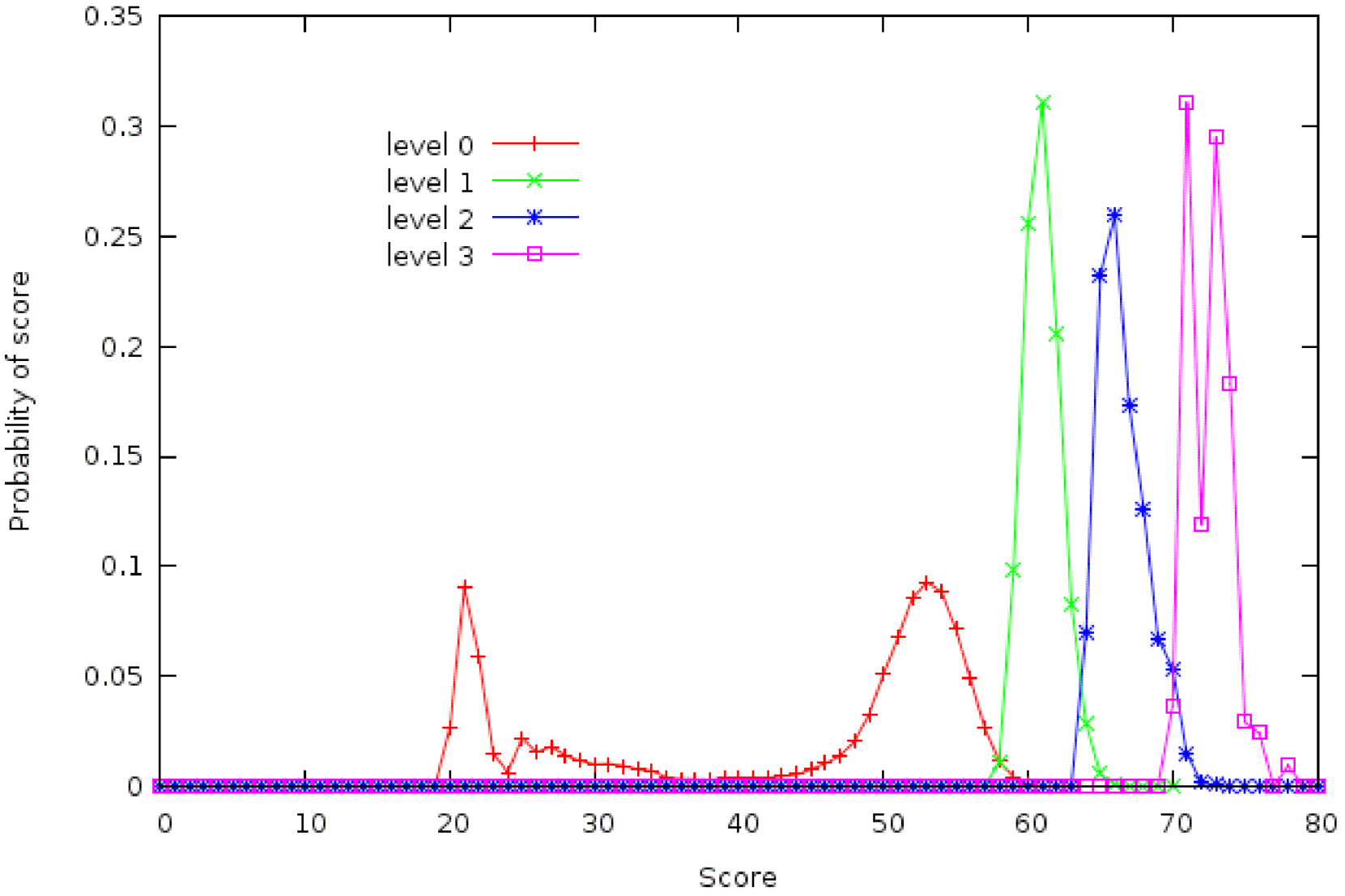
Morpion Solitaire

- Distribution of the scores without memorization



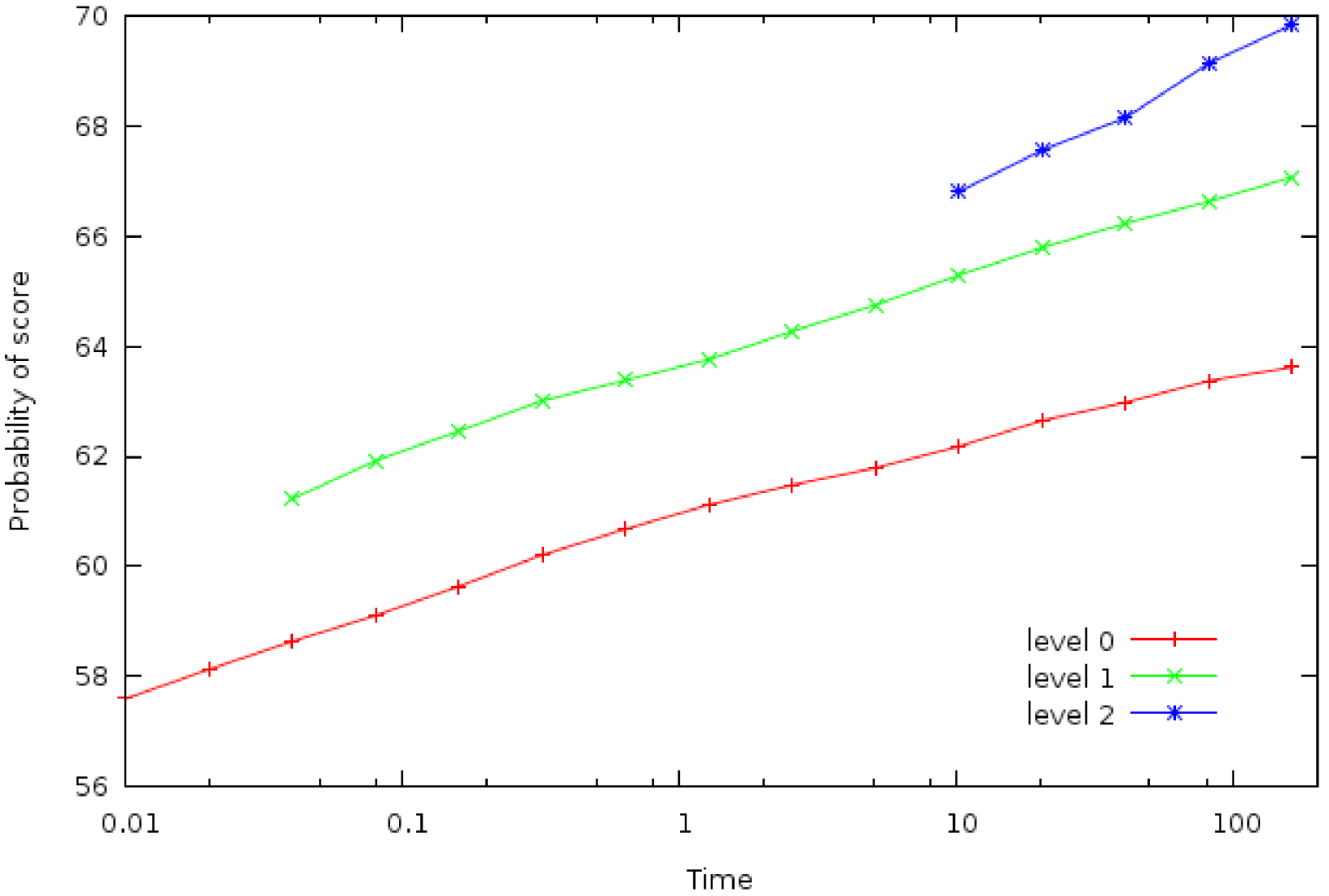
Morpion Solitaire

- Distribution of the scores with memorization



Morpion Solitaire

- Mean scores with memorization in real-time



Same Game

- NP-complete puzzle.
- It consists in a grid composed of cells of different colors. Adjacent cells of the same color can be removed together, there is a bonus of 1,000 points for removing all the cells.
- TabuColorRandom strategy: the color that has the most cells is set as the tabu color. During the playouts, moves of the tabu color are played only if there are no moves of the others colors.

Same Game

- SP-MCTS = restarts of the UCT algorithm
- SP-MCTS scored 73,998 on a standard test set.
- IDA* : 22,354
- Darse Billings program : 72,816.
- Nested level 3 : 77,934
- Nested level 2 with memorization : 65,937
- Level 2 without memorization : 44,731

Monte-Carlo CSP

- Search algorithms
  - Forward Checking
  - Iterative Sampling
  - Nested Monte-Carlo search
- Experiments on Kakuro and Sudoku

Forward Checking

- Reduce the set of possible values of the free variables after each assignment of a value to a given variable.
- In Kakuro : remove the value from the variable in the same column or row.
- The maximum value for a variable is also checked.
- Forward Checking = depth first search.
- More elaborate consistency checks are possible

Forward Checking

```
bool FC ()
  if no free variable then
    return true
  for all values in the domain of the variable
    assign value to variable
    if consistent then
      if FC () then
        return true
  return false
```

Iterative Sampling

- Playout = Randomly assign a value to a free variable while the assignment is consistent.

- Iterative Sampling = iteratively call the playout function while a solution is not found.

Iterative Sampling

```
1   int sample ()
2     while true
3       choose a free variable var
4       choose a value in the domain of var
5       assign value to var
6       update the domains of the free variables
7       if a domain is empty or inconsistent then
8         return 1 + number of free variables
9       if no free variable then
10        return 0
```

Meta Monte-Carlo search

- For each possible assignment of a value to the variable, plays a sample.
- Score of a sample = number of free variables when the playout is stopped.
- Choose the value that has the best sample score.
- Memorize the best sample so as to play it if no better sample has been found.

Meta Monte-Carlo Search

```
int meta (level)
  while true
    for all values in the domain of the variable
      assign value to variable
      score of value = sample ()
    assign the best found value to variable
    if end of game then
      return score
```

Nested Monte-Carlo Search

- A nested search of level 0 is a playout.
- A nested search of level 1 is a meta Monte-Carlo search.
- A nested search of level 2 uses nested search of level 1 to choose a value.
- etc.
- The score is always the number of free variables.

Nested Monte-Carlo Search

```
int nested (level)
  while true
    for all values in the domain of the variable
      assign value to variable
      if level is 1 then
        score = sample ()
      else
        score = nested (level - 1)
    assign the best found value to variable
    if end of game then
      return score
```

Kakuro

	24	25	20	26	24
18	.	.	.	.	.
26	.	.	.	.	.
28	.	.	.	.	.
26	.	.	.	.	.
21	.	.	.	.	.

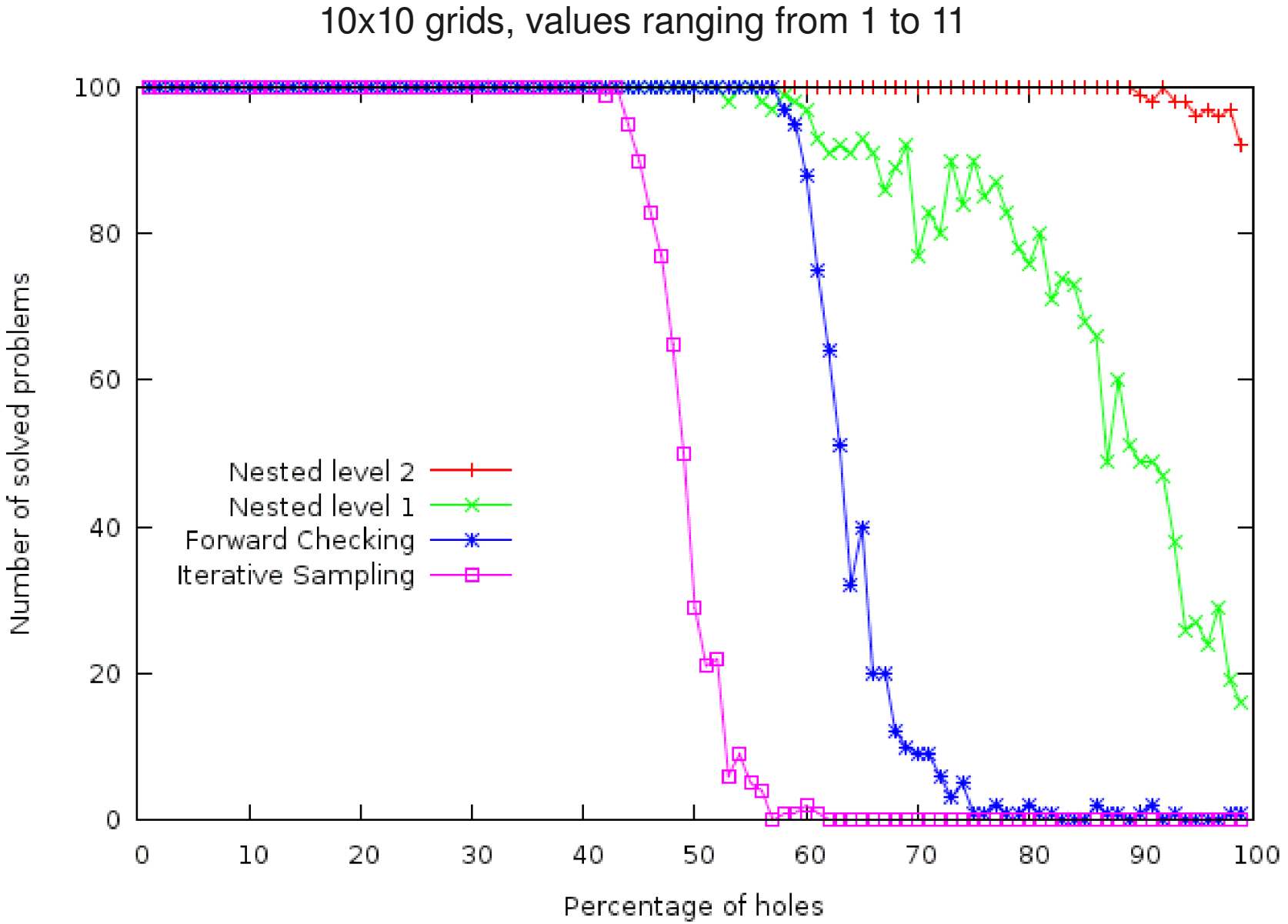
An empty 5x5 grid

Kakuro

	24	25	20	26	24
18	1	7	5	3	2
26	4	5	3	8	6
28	5	6	7	2	8
26	8	4	1	6	7
21	6	3	4	7	1

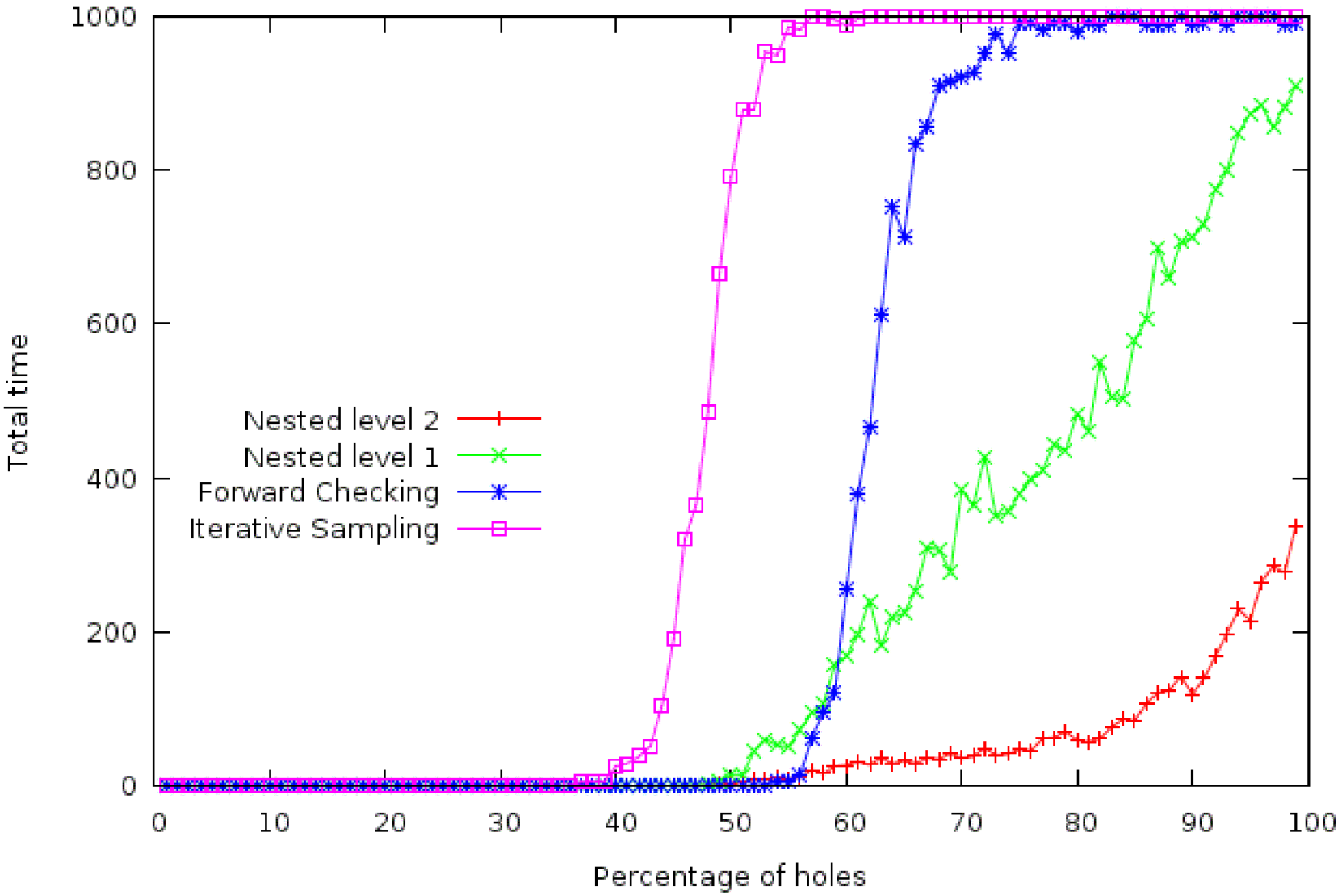
A solution to the previous grid

Experimental results



Experimental results

10x10 grids, values ranging from 1 to 11



Experimental results

Algorithm	Problems solved	Total time
Nested Level 2	100/100	0.91 s.
Nested Level 1	100/100	1.42 s.
Iterative Sampling	100/100	424.21 s.
Forward Checking	98/100	9,433.98 s.

Empty 6x6 grids, values in 1..7, timeout of 1,000 s.

Experimental results

Algorithm	Problems solved	Total time
Nested Level 2	100/100	17.85 s.
Nested Level 1	100/100	78.30 s.
Iterative Sampling	10/100	94,605.16 s.
Forward Checking	8/100	92,131.18 s.

Empty 8x8 grids, values in 1..9, timeout of 1,000 s.

Experimental results

Solved	Possible Values	Total time
100/100	[1,9]	17.85 s.
100/100	[1,10]	1,939.51 s.
100/100	[1,11]	1,166.13 s.
100/100	[1,12]	1,214.29 s.
99/100	[1,13]	1,688.51 s.
100/100	[1,14]	629.47 s.
100/100	[1,15]	672.90 s.
100/100	[1,16]	603.38 s.
100/100	[1,17]	429.26 s.
100/100	[1,18]	579.99 s.

Empty 8x8 grids , timeout of 1,000 s.

Sudoku

- Sudoku is a popular NP-complete puzzle.
- 16x16 grids with 66% of empty cells.
- Easy-Hard-Easy distribution of problems.
- Forward Checking (FC) is stopped when the search time for a problem exceeds 20,000 s.

Sudoku

- FC : > 446,771.09 s.
- Iterative Sampling : 61.83 s.
- Nested level 1 : 1.34 s.
- Nested level 2 : 1.64 s.
- A level 1 search without memorization of the best sequence takes 7.00 seconds instead of 1.34 seconds.
- A level 2 search without memorization takes 4.87 seconds instead of 1.64 seconds.

Bus Regulation

- Goal : minimize passengers waiting times at the bus stops by making buses wait at a stop.
- Algorithms :
  - Rule based approach.
  - Monte-Carlo.
  - Nested Monte-Carlo.

Rule-based Bus Regulation

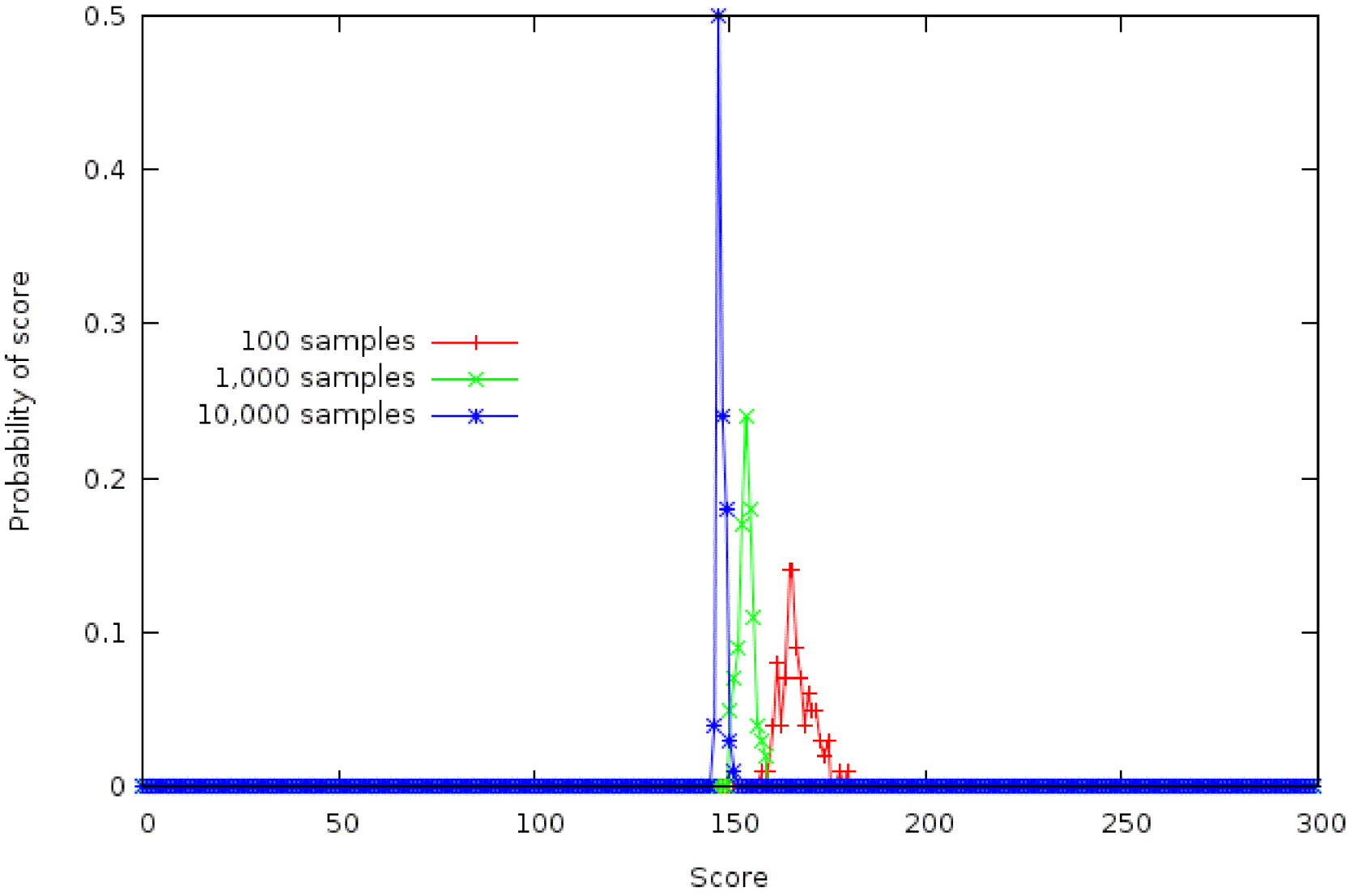
- Human regulators use rules of thumb to regulate the bus network in case of an incident, and this can be modeled with rules.
- Two values :
  - Delta : the threshold on the number of stops between the current bus and the following bus.
  - W : number of time steps the bus will wait at the current stop when the next bus is more than delta stops behind.

Rule-based Bus Regulation

	w=1	w=2	w=3	w=4
delta=4	171	192	193	199
delta=5	171	191	192	195
delta=6	171	175	176	198
delta=7	171	169	166	**164**
delta=8	171	169	167	165
delta=9	171	169	167	166
delta=10	171	170	170	170

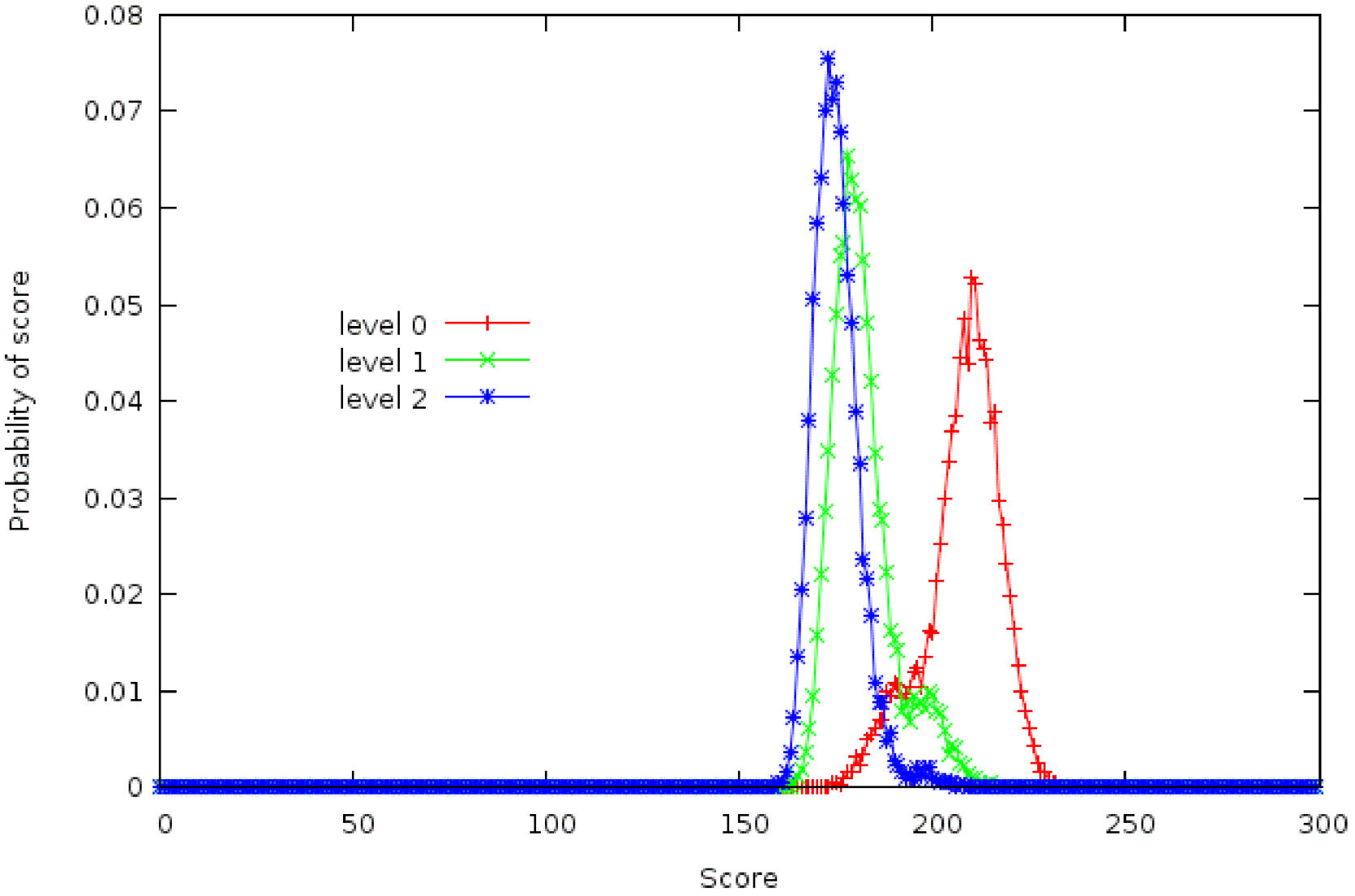
Monte-Carlo Bus Regulation

- Future is not known, randomized playouts :



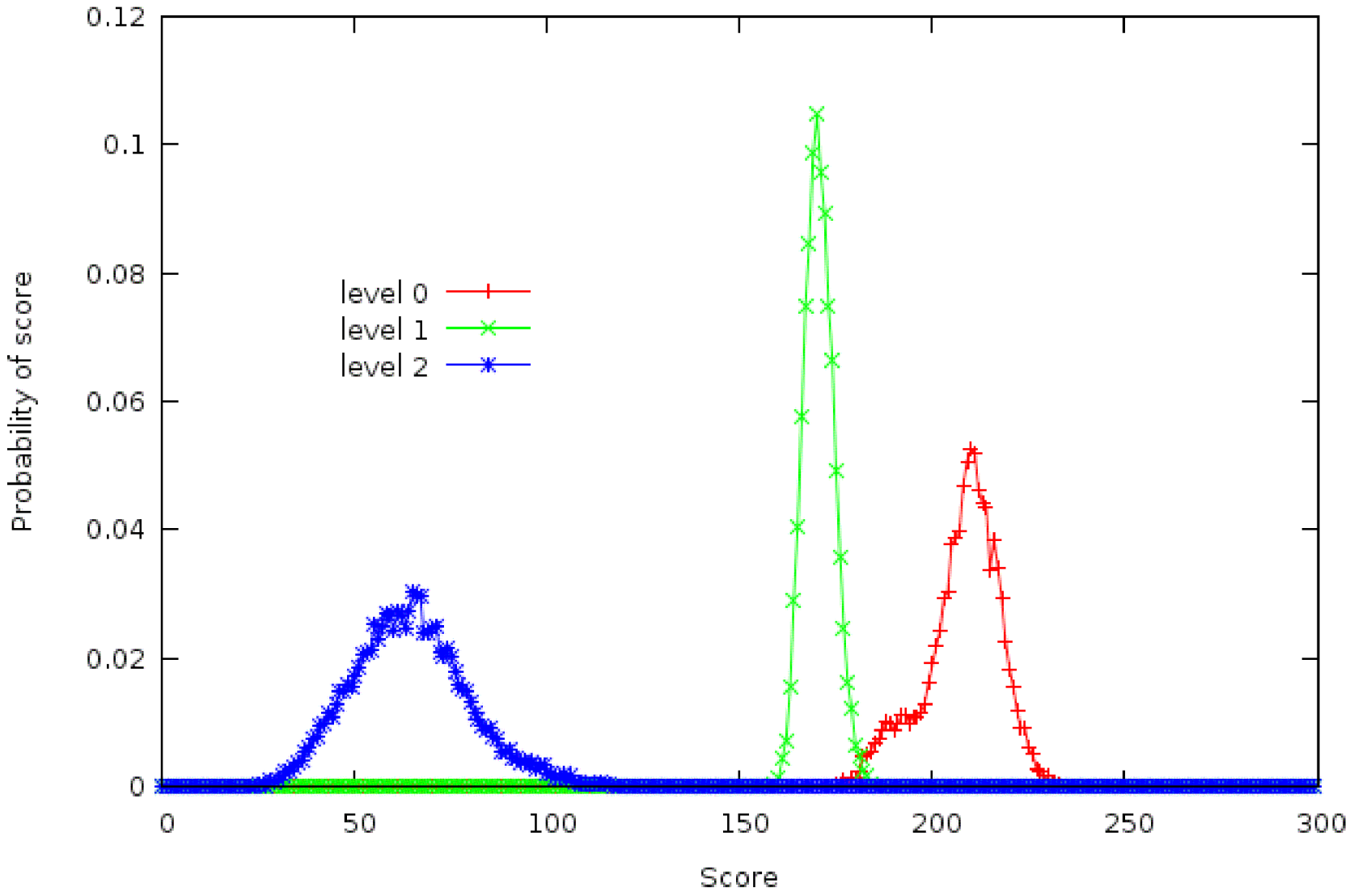
Nested Monte-Carlo Bus Regulation

- Future is known, no memorization :



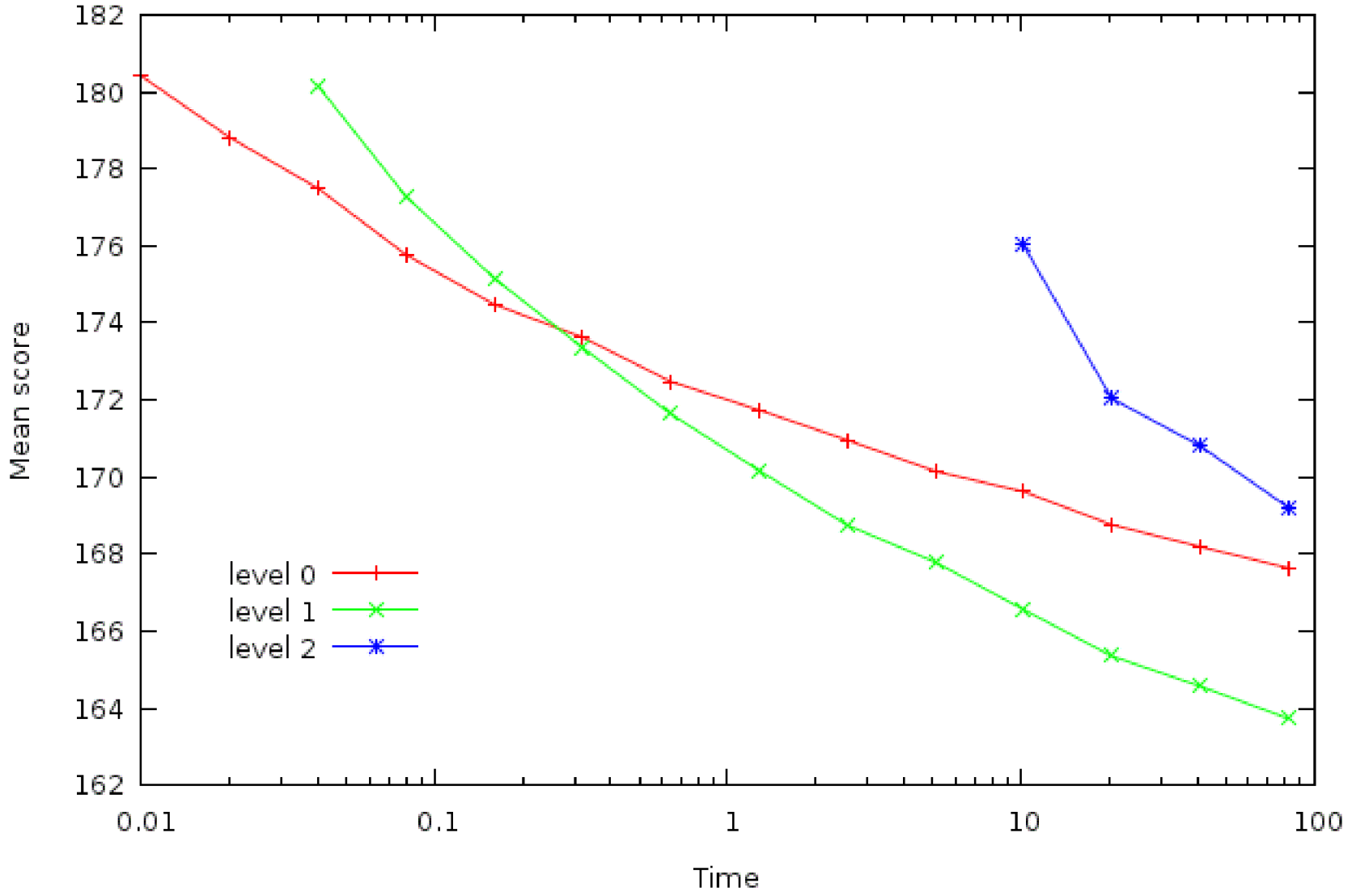
Nested Monte-Carlo Bus Regulation

- Future is known, memorize best sequence :



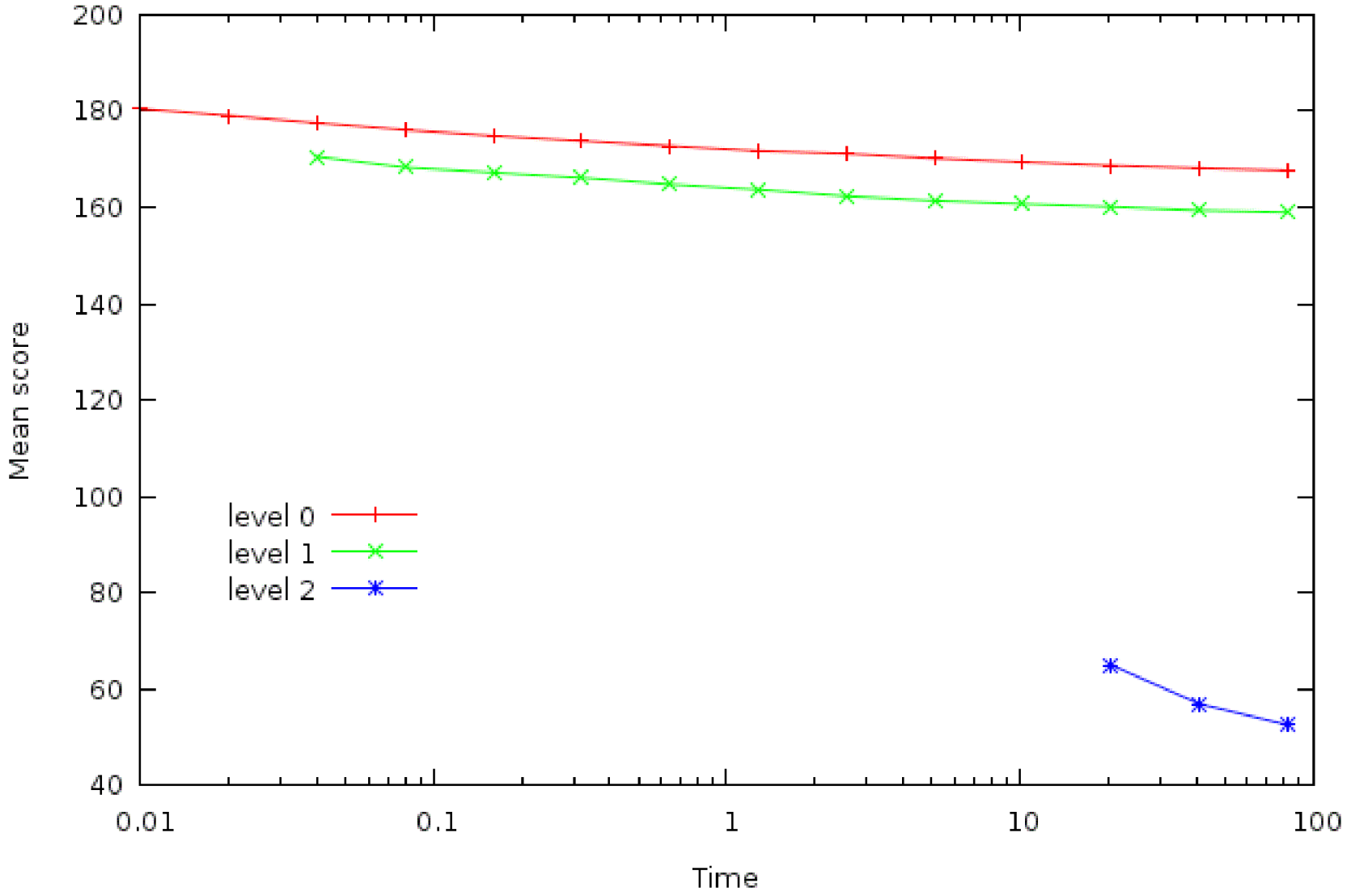
Nested Monte-Carlo Bus Regulation

- Mean scores, no memorization, real-time :



Nested Monte-Carlo Bus Regulation

- Mean scores, memorization, real-time :



Conclusion

- Memorizing the best sequence improves a lot Nested Monte-Carlo Search.
- Nested Monte-Carlo search found world records at Morpion Solitaire and Same Game.
- It can work for Constraint Satisfaction problems: Kakuro (level 2) and Sudoku (level 1).
- It can work for other problems such as bus regulation.