Improving Depth-first PN-Search: $1 + \varepsilon$ Trick

Jakub Pawlewicz    Łukasz Lew

Institute of Informatics
Warsaw University

Computer and Games 2006
PN-Search and DF-PN

1+ε Trick

Experiments
Relevant Work.

2004  Winands et al: PDS-PN.
Outline

1. PN-Search and DF-PN
2. Weak Point of DF-PN
3. 1+ε Trick
4. Experiments

Pawlewicz, Lew (Warsaw University) Improving Depth-first PN-Search: 1 + ε Trick Computer and Games 2006
PN-Search.

- PN-Search is AND/OR tree search algorithm.
- Uses proof and disproof numbers to find MPN.
- Iteratively expands Most Proving Node.

Recursive formula for the proof and disproof numbers:

\[
\begin{align*}
0 &\quad \text{proved} \\
1 &\quad \text{unknown}
\end{align*}
\]
PN-Search is AND/OR tree search algorithm.

- Uses proof and disproof numbers to find MPN.
- Iteratively expands Most Proving Node.

Recursive formula for the proof and disproof numbers:

\[
\begin{align*}
\text{proved} & : 0 + \infty \\
\text{unknown} & : 1
\end{align*}
\]
PN-Search is AND/OR tree search algorithm.

Uses proof and disproof numbers to find MPN.

Iteratively expands Most Proving Node.

Recursive formula for the proof and disproof numbers:

\[\begin{align*}
0 & \quad \text{proved} + \infty \\
1 & \quad \text{unknown}
\end{align*}\]
PN-Search is AND/OR tree search algorithm.
Uses proof and disproof numbers to find MPN.
Iteratively expands Most Proving Node.

Recursive formula for the proof and disproof numbers:

\[ \begin{align*}
p & = \text{proved} + \infty \\
d & = \text{unknown} + 1
\end{align*} \]
PN-Search is AND/OR tree search algorithm.
Uses proof and disproof numbers to find MPN.
Iteratively expands Most Proving Node.

Recursive formula for the proof and disproof numbers:

\[
\text{min}(p_1, p_2, p_3) \quad d_1 + d_2 + d_3
\]
PN-Search.

- PN-Search is AND/OR tree search algorithm.
- Uses proof and disproof numbers to find MPN.
- Iteratively expands Most Proving Node.

Recursive formula for the proof and disproof numbers:

\[
\begin{align*}
0 & \text{ proved} + \infty \\
1 & \text{ unknown}
\end{align*}
\]

\[
\begin{align*}
\text{OR} & \quad d_1 + d_2 + d_3 \\
\text{AND} & \quad \min(p_1, p_2, p_3) \\
p_1 & \quad d_1 \\
p_2 & \quad d_2 \\
p_3 & \quad d_3
\end{align*}
\]
DF-PN is a PN-Search transformation to depth-first algorithm.

- Suspend updates as long as MPN is in current node’s subtree.
- Uses proof and disproof number thresholds.
- Return condition: $p \geq pt \lor d \geq dt$.

Calculating the thresholds:

$\begin{align*}
\text{root} & = +\infty + +\infty \\
\text{AND} & \quad \text{AND} \\
\quad p_1 & \quad p_2 \\
\quad d_1 & \quad d_2 \\
\quad \text{OR} & \quad \text{AND} \\
\quad p & \quad d \\
\quad pt & \quad dt \\
\quad p_3 & \\
\quad d_3 & \\
\quad \text{AND} & \\
\quad p_1 \leq p_2 \leq p_3 & \\
\end{align*}$
DF-PN is a PN-Search transformation to depth-first algorithm.
Suspend updates as long as MPN is in current node’s subtree.
Uses proof and disproof number thresholds.
Return condition: $p \geq pt \lor d \geq dt$.

Calculating the thresholds

$\begin{align*}
\text{root} + \infty & \quad + \infty \\
\land p_1 d_1 \quad \lor \quad p_2 d_2 \quad \land p_3 d_3 \\
\text{OR} p & \quad d \\
\land p_1 \leq p_2 \leq p_3
\end{align*}$
DF-PN

- DF-PN is a PN-Search transformation to depth-first algorithm.
- Suspend updates as long as MPN is in current node’s subtree.
- Uses proof and disproof number thresholds.
- Return condition: \( p \geq pt \lor d \geq dt \).

Calculating the thresholds

\[ \begin{align*}
\text{OR} & \quad \text{AND} \\
p & \quad d \\
p_1 \quad & \quad d_1 \\
p_2 \quad & \quad d_2 \\
p_3 \quad & \quad d_3
\end{align*} \]

\[ p_1 \leq p_2 \leq p_3 \]
DF-PN is a PN-Search transformation to depth-first algorithm. Suspend updates as long as MPN is in current node’s subtree. Uses proof and disproof number thresholds. Return condition: \( p \geq pt \lor d \geq dt \).

Calculating the thresholds

\[ p_1 \leq p_2 \leq p_3 \]

\[ p_1 \land d_1 \lor p_2 \land d_2 \land p_3 \land d_3 \]
DF-PN is a PN-Search transformation to depth-first algorithm.

- Suspend updates as long as MPN is in current node’s subtree.
- Uses proof and disproof number thresholds.
- Return condition: \( p \geq pt \lor d \geq dt \).

Calculating the thresholds

\[ p_1 \leq p_2 \leq p_3 \]
DF-PN is a PN-Search transformation to depth-first algorithm.
Suspend updates as long as MPN is in current node’s subtree.
Uses proof and disproof number thresholds.
Return condition: $p \geq pt \lor d \geq dt$.

Calculating the thresholds

\[
\min(p_2 + 1, pt) \quad d_1
\]

\[
p_1 \quad \text{AND} \quad dt - d + d_1
\]

\[
p_2 \quad \text{AND} \quad d_2
\]

\[
p_3 \quad \text{AND} \quad d_3
\]

$p_1 \leq p_2 \leq p_3$
1 PN-Search and DF-PN

2 Weak Point of DF-PN

3 $1+\varepsilon$ Trick

4 Experiments
Typical Situation During a Run of DF-PN.

OR

750

900
Typical Situation During a Run of DF-PN.

PN History

300 302
305 306
308 311
315 317
...
742 746
750 752

Pawlewicz, Lew (Warsaw University)
Improving Depth-first PN-Search: 1 + ε Trick
Computer and Games 2006
Typical Situation During a Run of DF-PN.

Pawlewicz, Lew (Warsaw University)  Improving Depth-first PN-Search: $1 + \varepsilon$ Trick  Computer and Games 2006  8 / 15
Typical Situation During a Run of DF-PN.

PN History

300 302 306 311 317 ...
742 746 750 752
Typical Situation During a Run of DF-PN.
Typical Situation During a Run of DF-PN.

PN History

300 302 306 311 317 ...
742 746 750 752

Pawlewicz, Lew (Warsaw University)  Improving Depth-first PN-Search: 1 + $\epsilon$ Trick  Computer and Games 2006
Typical Situation During a Run of DF-PN.
Typical Situation During a Run of DF-PN.

Pawlewicz, Lew (Warsaw University)  
Improving Depth-first PN-Search: $1 + \varepsilon$ Trick  
Computer and Games 2006 8 / 15
Typical Situation During a Run of DF-PN.

PN History

300 302 306 311 317 ...
742 746 750 752
Typical Situation During a Run of DF-PN.

Pawlewicz, Lew (Warsaw University) Improving Depth-first PN-Search: 1 + $\epsilon$ Trick Computer and Games 2006 8 / 15
Typical Situation During a Run of DF-PN.

PN History

300 302
305 306
308 311
315 317
...
742 746
750 752
Typical Situation During a Run of DF-PN.
Typical Situation During a Run of DF-PN.

PN History

300 302 306 311 317 ...
742 746 750 752
Typical Situation During a Run of DF-PN.

Pawlewicz, Lew (Warsaw University)  Improving Depth-first PN-Search: $1 + \varepsilon$ Trick  Computer and Games 2006 8 / 15
Typical Situation During a Run of DF-PN.

PN History:
- 300
- 302
- 306
- 311
- 317
- ...
- 742
- 746
- 750
- 752

Diagram:
- AND node with values 307 and 214
- OR node with values 750 and 900
- AND node with values 306 and 106
- TT node with values 694

Pawlewicz, Lew (Warsaw University)

Improving Depth-first PN-Search: 1 + ε Trick

Computer and Games 2006
Typical Situation During a Run of DF-PN.

Pawlewicz, Lew (Warsaw University)  Improving Depth-first PN-Search: $1 + \varepsilon$ Trick  Computer and Games 2006  8 / 15
Outline

1. PN-Search and DF-PN
2. Weak Point of DF-PN
3. $1 + \varepsilon$ Trick
4. Experiments

Pawlewicz, Lew (Warsaw University)  Improving Depth-first PN-Search: $1 + \varepsilon$ Trick  Computer and Games 2006
Advantages

- Reduced number of recursive calls and transposition table refills.
- Less tree traversal.

Disadvantages

- Leaves are not expanded in the same order as in PN-Search.
- For bigger $\varepsilon$ the algorithm may spend too much time in inessential part of a tree.
1 + \( \varepsilon \) Trick.

\[
\min(p_2(1+\varepsilon), pt) \leq p \leq p_1 \quad \text{AND} \quad dt - d + d_1 \leq d_2 \quad \text{OR} \\
\]

Advantages
- Reduced number of recursive calls and transposition table refills.
- Less tree traversal.

Disadvantages
- Leaves are not expanded in the same order as in PN-Search.
- For bigger \( \varepsilon \) the algorithm may spend too much time in inessential part of a tree.
1 + $\varepsilon$ Trick.

Advantages
- Reduced number of recursive calls and transposition table refills.
- Less tree traversal.

Disadvantages
- Leaves are not expanded in the same order as in PN-Search.
- For bigger $\varepsilon$ the algorithm may spend too much time in inessential part of a tree.
$1 + \varepsilon$ Trick.

$$\min([p_2(1+\varepsilon)], pt)$$

$\min(p_1, \min([p_2(1+\varepsilon)], pt))$

Advantages

- Reduced number of recursive calls and transposition table refills.
- Less tree traversal.

Disadvantages

- Leaves are not expanded in the same order as in PN-Search.
- For bigger $\varepsilon$ the algorithm may spend too much time in inessential part of a tree.
1 + $\varepsilon$ Trick.

\[ \min([p_2(1+\varepsilon)], pt), dt - d + d_1 \]

Advantages
- Reduced number of recursive calls and transposition table refills.
- Less tree traversal.

Disadvantages
- Leaves are not expanded in the same order as in PN-Search.
- For bigger $\varepsilon$ the algorithm may spend too much time in inessential part of a tree.
Outline

1. PN-Search and DF-PN
2. Weak Point of DF-PN
3. $1+\varepsilon$ Trick
4. Experiments

Pawlewicz, Lew (Warsaw University) Improving Depth-first PN-Search: $1+\varepsilon$ Trick Computer and Games 2006
Atari Go: The Size of a Transposition Table

- Enhanced DF-PN
- DF-PN
- Enhanced PDS
- PDS

solving time in seconds vs. transposition table size in number of nodes
Atari Go: The Size of a Transposition Table

transposition table size in number of nodes

solving time in seconds

enhanced DF-PN

enhanced PDS

DF-PN

PDS

Pawlewicz, Lew (Warsaw University)

Improving Depth-first PN-Search: $1 + \varepsilon$ Trick

Computer and Games 2006
Atari Go: The Size of a Transposition Table

Pawlewicz, Lew (Warsaw University)  
Improving Depth-first PN-Search: $1 + \varepsilon$ Trick  
Computer and Games 2006
LOA: Efficiency of Solving Hard Problems

Average solving times in seconds

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>enhanced DF-PN</td>
<td>45</td>
</tr>
<tr>
<td>DF-PN</td>
<td>80</td>
</tr>
<tr>
<td>enhanced PDS</td>
<td>241</td>
</tr>
<tr>
<td>PDS</td>
<td>328</td>
</tr>
</tbody>
</table>

Pawlewicz, Lew (Warsaw University)  
Improving Depth-first PN-Search: 1 + ε Trick  
Computer and Games 2006  
13 / 15
LOA: Efficiency of Solving Hard Problems

Average solving times in seconds

<table>
<thead>
<tr>
<th>Method</th>
<th>Average Solving Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>enhanced DF-PN</td>
<td>45</td>
</tr>
<tr>
<td>DF-PN</td>
<td>80</td>
</tr>
<tr>
<td>enhanced PDS</td>
<td>241</td>
</tr>
<tr>
<td>PDS</td>
<td>328</td>
</tr>
</tbody>
</table>

Enhanced DF-PN vs. DF-PN vs. PDS

Pawlewicz, Lew (Warsaw University) Improving Depth-first PN-Search: $1 + \varepsilon$ Trick Computer and Games 2006
LOA: Efficiency of Solving Hard Problems

Average solving times in seconds

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>enhanced DF-PN</td>
<td>45</td>
</tr>
<tr>
<td>DF-PN</td>
<td>80</td>
</tr>
<tr>
<td>enhanced PDS</td>
<td>241</td>
</tr>
<tr>
<td>PDS</td>
<td>328</td>
</tr>
</tbody>
</table>

Enhanced DF-PN, DF-PN, enhanced PDS, PDS

Positions solved out of 286 vs. running time limit

Pawlewicz, Lew (Warsaw University) Improving Depth-first PN-Search: $1 + \varepsilon$ Trick

Computer and Games 2006 13 / 15
LOA: Efficiency of Solving Hard Problems

Average solving times in seconds

<table>
<thead>
<tr>
<th></th>
<th>enhanced DF-PN</th>
<th>DF-PN</th>
<th>enhanced PDS</th>
<th>PDS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>45</td>
<td>80</td>
<td>241</td>
<td>328</td>
</tr>
</tbody>
</table>

Pawlewicz, Lew (Warsaw University) Improving Depth-first PN-Search: 1 + ε Trick Computer and Games 2006 13 / 15
Summary

- We have pointed out the problems in DF-PN.
- We have introduced $1 + \varepsilon$ trick to enhance DF-PN.
- Atari Go experiment has shown that enhanced methods outperform their plain variants in low memory conditions.
- Experiment on LOA has shown that the trick is also valuable for solving hard problems.
Thank You

Questions?