Proof for the Equivalence Between Some Best-First Algorithms and Depth-First Algorithms for AND/OR Trees

Ayumu Nagai and Hiroshi Imai
Department of Information Science
University of Tokyo
E-mail: nagai@is.s.u-tokyo.ac.jp

Overview

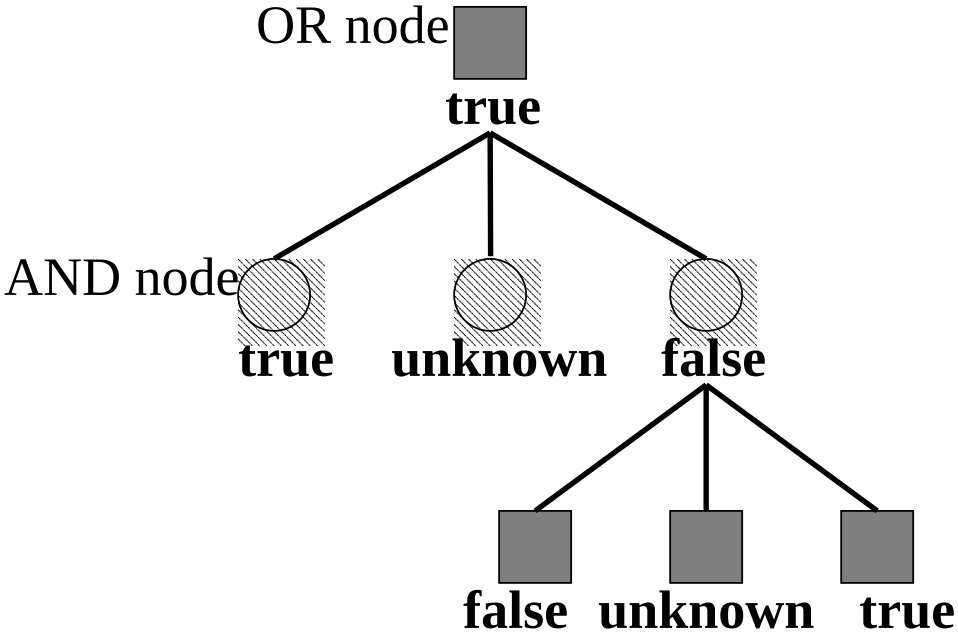
- Explanation of an AND/OR tree
  - Proof number and disproof number
  - Most-proving node
- Introduction of some algorithms
  - Pn-search
  - Df-pn-search
- Proof the outline of the equivalence
  - In the meaning of expanding always a most-proving node
- Experimental results
- Conclusion

An AND/OR Tree

- Two players $\cdots$ first player and second player
- Three values $\cdots$ false, unknown, and true
- Ordering (at OR nodes) $\cdots$ false $<$ unknown $<$ true
- Ordering (at AND nodes) $\cdots$ true $<$ unknown $<$ false

The final value of the root

- true $\Rightarrow$ proof solution (proved)
- false $\Rightarrow$ disproof solution (disproved)



Proof Number and Disproof Number

- If e is proved

  $\Downarrow$

  a is proved

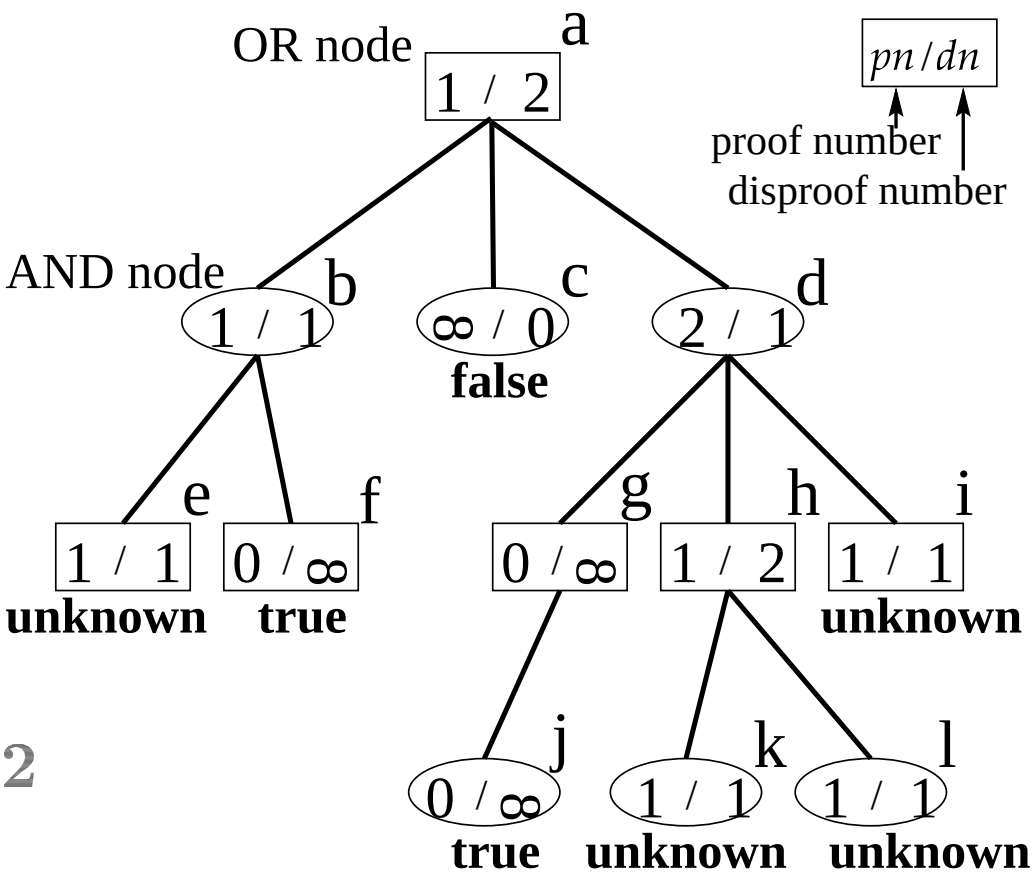
  $\Longrightarrow$ Proof number of a $= 1$

- If e and i is disproved

  $\Downarrow$

  a is disproved

  $\Longrightarrow$ Disproof number of a $= 2$



- e is the most-proving node

  $\Longrightarrow$ Affecting to both proof number and disproof number

Calculation of Proof Number and Disproof Number

- **At an OR node n**
  - **One of its children is proved $\Longrightarrow$ proved**
  - **All the children is disproved $\Longrightarrow$ disproved**

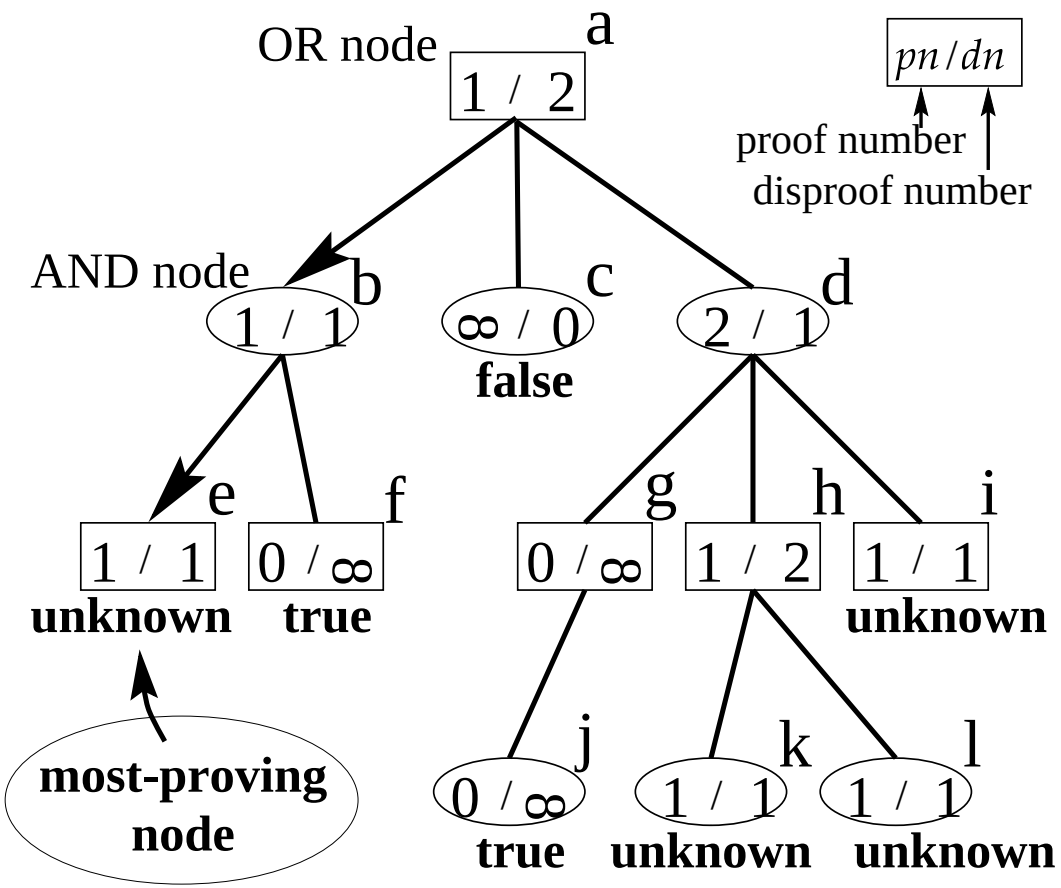
$$n.pn = \min_{n_{\text{child}} \in \textbf{children of } n} n_{\text{child}}.pn \quad \Longleftarrow$$
$$n.dn = \sum_{n_{\text{child}} \in \textbf{children of } n} n_{\text{child}}.dn$$

- **At an AND node n**
  - **All the children is proved $\Longrightarrow$ proved**
  - **One of its children is disproved $\Longrightarrow$ disproved**

$$n.pn = \sum_{n_{\text{child}} \in \textbf{children of } n} n_{\text{child}}.pn$$
$$n.dn = \min_{n_{\text{child}} \in \textbf{children of } n} n_{\text{child}}.dn$$

A Most-Proving Node

- Starting from the root
- Selection among the child
  - At an OR node
    - ··· child of least proof number
  - At an AND node
    - ··· child of least disproof number



$\Longrightarrow$ Finally reaches to the most-proving node

Related Work and its Classification

	criteria of evaluation used	
	only proof number	proof number and disproof number
best-first	AO*[Nilson,1980] (Elkan[1989])	pn-search[Allis,1994]
depth-first	Seo's Algorithm [1995]	df-pn-search[this paper] (PDS[Nagai,1998])

Purpose

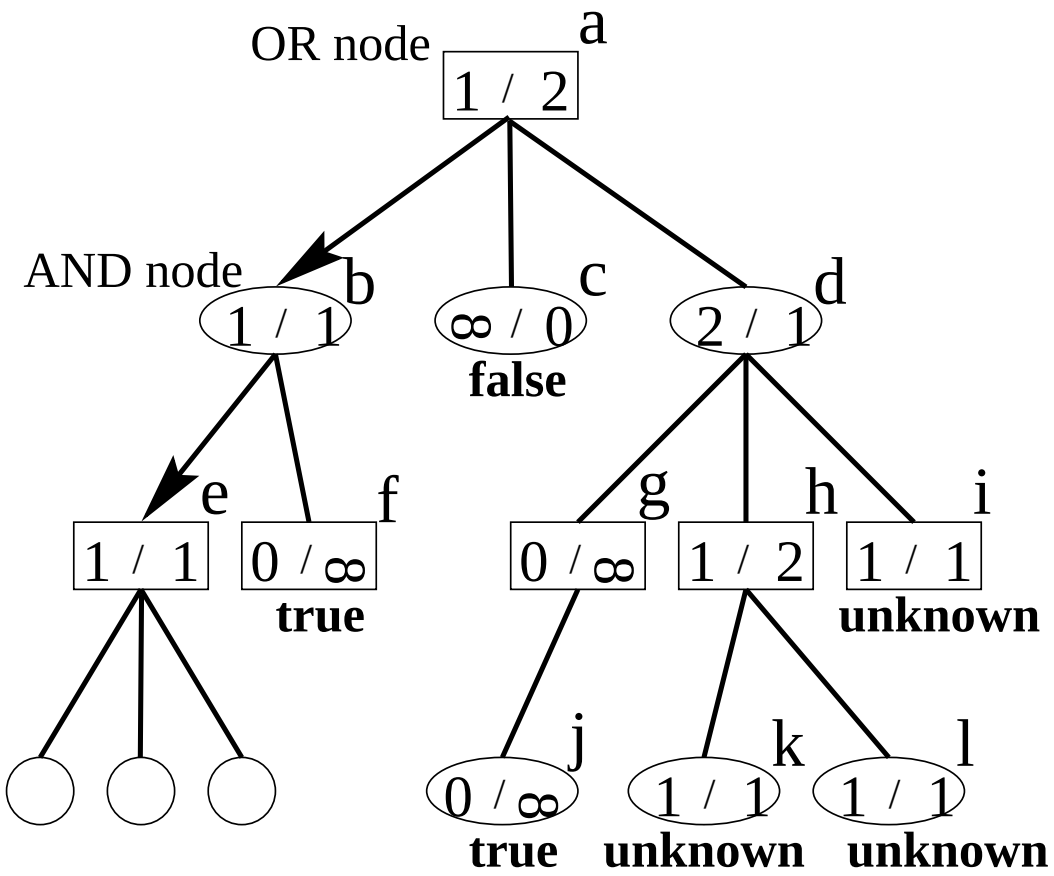
- Proof of the equivalence
  - between AO* and Seo's Algorithm
  - between pn-search and df-pn-search $\Leftarrow$ focus

Pn-search

1. Select a most-proving node
2. Generate all the children
3. Propagate toward the root

Repeat until
the solution is found

$\Longrightarrow$ Goes back and forth

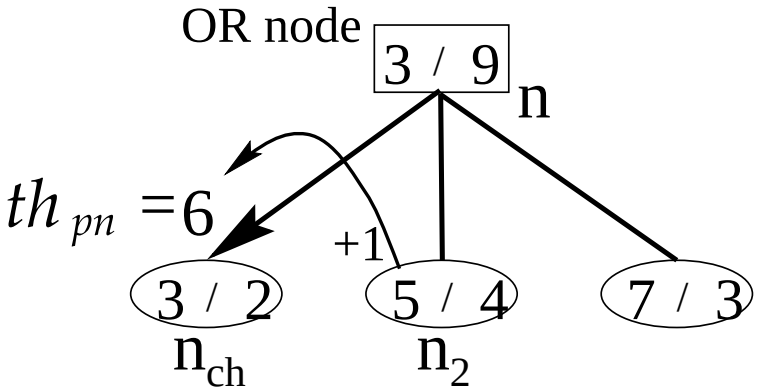


Df-pn-search

- **At each node n, search below n**

  while $n.pn < n.th_{pn}$

  and $n.dn < n.th_{dn}$



- **At each OR node n,**
  - **search the child n_{ch} with minimum proof number**
  - **assign**

$$n_{\mathrm{ch}}.th_{pn} = \mathbf{min}(n.th_{pn}, \underline{n_2.pn + 1})$$

$$n_{\mathrm{ch}}.th_{dn} = n.th_{dn} + n_{\mathrm{ch}}.dn - \Sigma n_{\mathrm{child}}.dn$$

$n_2 \cdots$ **is the child with second minimum proof number**

Proof of the equivalence (1)

(!(! Course of the Proof (!(!

- Pn-search always expands most-proving node

$\Uparrow$

self-evident

- Df-pn-search always expands most-proving node

$\Downarrow$

proof outline

$\Longrightarrow$ pn-search and df-pn-search is equivalent

Proof of the equivalence (2)

(!(! Expansion of Most-Proving Node during Df-pn-search (!(!

A leaf node is now to be expanded

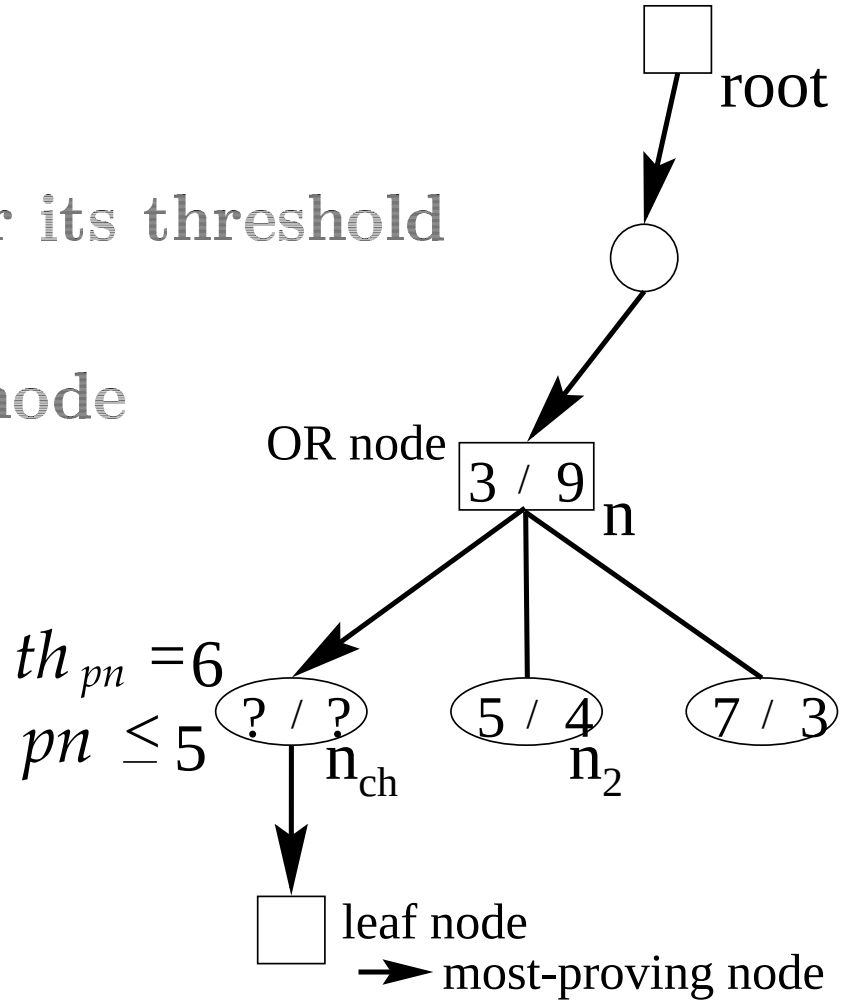
- While (dis)proof number is under its threshold

$\Downarrow$

Most-proving node is below the node

- The same thing can be said with all the active nodes

$\Longrightarrow$ Df-pn-search always expands most-proving node



Proof of the equivalence (3)

(!(! More Strong Equivalence (!(!

- A narrow definition of most-proving node

$\Longrightarrow$ Most-proving node is defined uniquely
(but dynamically)

$\Longrightarrow$ Same most-proving nodes are expanded in the same order
as with both pn-search and df-pn-search

Difference Between Pn-Search and Df-Pn-Search

- Pn-search searches

  $a \rightarrow b \rightarrow e \rightarrow b \rightarrow a \rightarrow b \rightarrow e \rightarrow m$

- Df-pn-search searches

  $a \rightarrow b \rightarrow e \rightarrow m$

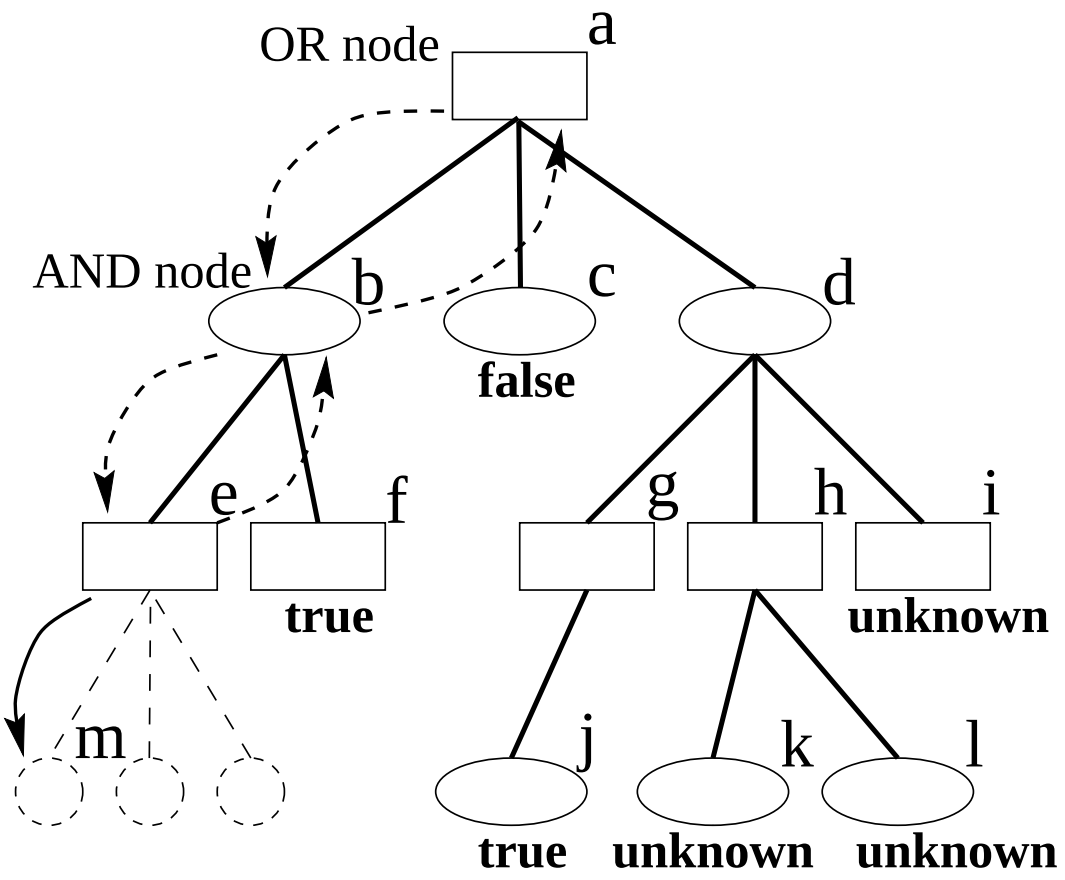
As with df-pn-search,

- No unnecessary

  back and forth

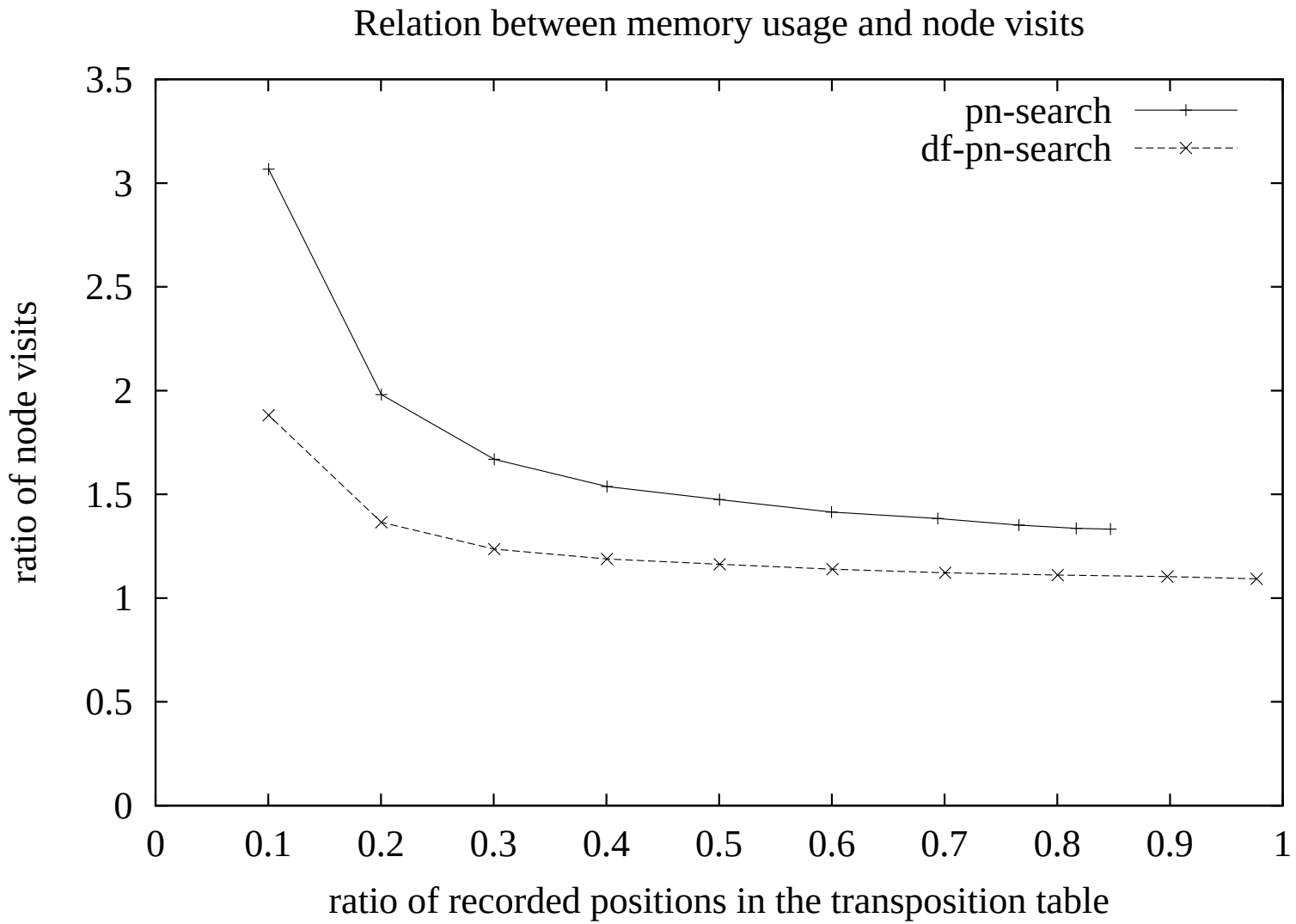
  $\Uparrow$

  Two thresholds



Experimental Result

(151 Othello positions with 15 vacant squares appeared at PrincetonII)



$\Longrightarrow$ Df-pn-search needs only 60-85% node visits

Conclusion

- Proposal of a new depth-first algorithm (df-pn-search)
- Proof of the equivalence
  between pn-search and df-pn-search
- Experimental result show the superiority of
  df-pn-search to pn-search

Future Work

- Improvement of Seo's algorithm
- Using the information of heuristic evaluation function