Conquering the Busy Beaver
presented by Kyle Ross
4th December 2002

Bram van Heuveln
Boleshaw Szymanski
Selmer Bringsjord
Carlos Varela
Owen Kellett
Shailesh Kelkar
Kyle Ross
Turing Machines

---

00000000000000000000000000000000---

---

Gregg's Challenge

start

0

1

0:1 1:R

1:R

0:1 0:0

0:1

2

3

4

5

0:1 1:L

1:L

1:L
Turing Machines

Gregg's Challenge
Turing Machines

Gregg's Challenge

---
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 --

---

start

0

1:R

0:1

1:R

0:0

0:1

1:L

1:L

0:1

0:1

1:L

1:L

3

4
Turing Machines

0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0

Gregg's Challenge
Turing Machines

Gregg's Challenge
Turing Machines

Gregg's Challenge

0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
The Busy Beaver Problem

“Consider, for a fixed positive integer \( n \), the class \( K_n \) of all the \( n \)-card [state] binary turing machines ... Let \( M \) be a Turing machine in this class \( K_n \). Start \( M \), with its card 1, on an all-0 tape. If \( M \) stops after a while, then \( M \) is termed a valid entry in the BB-\( n \) contest ... and its score \( \sigma(M) \) is the number of 1's remaining on the tape at the time it stops ... [the set of \( \sigma \)-values] has a (unique) largest element which we denote by \( \Sigma(n) \) ... It is practically trivial that this function \( \Sigma(n) \) is not general recursive ... [but] it may be possible to determine the value of \( \Sigma(n) \) for particular values of \( n \).”

Variants of the Problem

- quadruple vs. quintuple
- standard position vs. arbitrary format output
- implicit vs. explicit halt machine
Turing Machine Formulations

quintuple formulation

quadruple formulation
Turing Machine Formulations

explicit halt

implicit halt
Turing Machine Formulations

B(5)-11
Previous Work on Quadruple

- $R(n)$ - quadruple, explicit, no restriction
  - [nobody?]
- $O(n)$ - quadruple, implicit, no restriction
  - Oberschelp et al.
- $P(n)$ - quadruple, explicit, standard
  - Pereira et al.
- $B(n)$ - quadruple, implicit, standard
  - Boolos and Jeffrey
### Known Results

<table>
<thead>
<tr>
<th>n</th>
<th>R(n)</th>
<th>O(n)</th>
<th>P(n)</th>
<th>B(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>&gt;=2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>&gt;=4</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>&gt;=8</td>
<td>8</td>
<td>&gt;=7</td>
<td>&gt;=5</td>
</tr>
<tr>
<td>5</td>
<td>&gt;=16</td>
<td>15</td>
<td>&gt;=16</td>
<td>&gt;=11</td>
</tr>
<tr>
<td>6</td>
<td>&gt;=71</td>
<td>&gt;=70</td>
<td>&gt;=41</td>
<td>&gt;=25</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>&gt;=164</td>
<td>&gt;=164</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>&gt;=384</td>
<td></td>
</tr>
</tbody>
</table>
About the Quadruple Formulation

- Turing's World & Greg's challenge
- less-productive than quintuple machines
- greater room for optimisations
The Search Space

- $|M(n)| = (4n+1)^{2n}$
  - 4 possible actions for each of $n$ next states
  - 1 no-action transition to halt-state
  - $2n$ possible transitions
- for $B(6) = (4(6)+1)^{2(6)} = 5.96 \times 10^{16}$ machines
- not hopeless!
Inefficiency: Isomorphisms

B(5)-11

B(5)-11-isomorph
Inefficiency: Unused Transitions

B(4)-5-u1

B(4)-5-u2
Solution: Tree Normalisation
Solution: Tree Normalisation
Solution: Tree Normalisation

non-halter

halt
Solution: Tree Normalisation
Solution: Tree Normalisation
Features of Tree Normalisation

- complete & optimal search
- no loss of absolute numbers
- great speed-up over pure brute-force
Improvement from Normalisation

Tree Normalisation Improvement

reduction (in percentage of machines analysed)

n (number of states)
Inefficiency: Empty Tape Machine

- machine reaches an empty tape after 1 or more shifts
- any machine that does not write 1 as its first action is such a machine
(Partial) Solution: Force First Write

non-halter
first write not 1
(Partial) Solution: Force First Write
Improvement from First Write

Optimisations Improvement

reduction (in percentage of machines analysed)

n (number of states)

Tree Normalisation

First-Write Optimisation
Inefficiency: Nonproductive Transitions
Inefficiency: Mirror Machines

B(5)-11

B(5)-11-mirror
Solution: Force First Move
Solution: Force First Move
Solution: Force First Move
Solution: Force First Move

low-productivity
Solution: Force First Move
Improvement from First Move

Optimisations Improvement

reduction (in percentage of machines analysed)

n (number of states)

- Tree Normalisation
- First-Write Optimisation
- First-Move Optimisation
Distributed Computing

- still a lot of work to do (particularly for n>6)
- C/C++ farmer / worker model
- SALSA actor / theatre model
C++ Farmer / Worker Distribution

Farmer

Workers

8 levels

4 levels
SALSA Actor Distribution

Farmer

Sub-Farmers
Features of Farmer / Worker

- centralised view of problem
- dynamic search-space sub-division
- compatible with optimisations
- representation and partial machines
Future of the Problem

- ultimately will always remain non-computable
- always able to get candidates
- reduction to halting problem and limits of human analysis
- Kyle's prediction: nobody will get past B(8) for a very long time
RPI B(6) Champion*

* This machine is also the world champion (and probably the theoretical B(6) champion).
We Beat the Portuguese!

P(6)-41
We Have Records for O(6) & R(6)!