Proceedings of the Ninth International
Symposium on Combinatorial Search (SoCS 2016)

Burnt Pancake Problem: New Lower Bounds on the Diameter and New Experimental Optimality Ratios

Bruno Bouzy
LIPADE, Université Paris Descartes, FRANCE,
bruno.bouzy@parisdescartes.fr

Abstract

For the burnt pancake problem, we provide new values of $g(-I_N)$, new hard positions and new experimental optimality ratios.

Introduction

The pancake problem is well-known in computer science (Gates and Papadimitriou 1979). p being a given problem, $g(p)$ is the length of optimal solutions of p. N is the size of a pancake problem. $g(N)$ is the diameter of the graph corresponding to size N pancake problems. For the burnt pancake problem, $3N/2$ is a lower bound on $g(N)$ and $2(N-1)$ a upper bound (Cohen and Blum 1995). Beside, Cohen and Blum's algorithm (1995) is a 2-approximation algorithm. $-I_N$ is a known hard problem. (Cohen and Blum 1995) conjectured that $g(-I_N) = g(N)$.

First, while previous values on $g(-I_N)$ were known for $N \leq 20$ (Cibulka 2011), we provide new values for $N \leq 27$. This result is obtained with IDA* (Korf 1985) and the number of breakpoints (Gates and Papadimitriou 1979) as heuristic function. Secondly, for $N \leq 22$, we used a new heuristic function to uncover some hard positions, different from $-I_N$. Thirdly, we give experimental optimality ratios obtained with Monte-Carlo Search (MCS) (Cazenave 2009) using Cohen and Blum's algorithm for $N \leq 256$.

Definitions

Let $s = [s(1), s(2), ..., s(N-1), s(N)]$ be a stack of burnt pancakes. A burnt pancake is burnt on one side. $|s(i)|$ is the size of the pancake situated at position i. The sign of $s(i)$ corresponds to the orientation of the burnt side of pancake i. The burnt pancake problem consists in reaching the identity stack I_N such that $I_N(i) = i$ by applying a sequence of flips. A flip transforms s into $[\underline{-s(i), ..., -s(1)}, s(i+1), ..., s(N)]$. In the burnt pancake problem, a breakpoint is situated between i and $i-1$ when $s(i) - s(i-1) \neq 1$. $\#bp$ denotes the number of breakpoints. $\#bp$ is a lower bound of the length of optimal solutions (Gates and Papadimitriou 1979), (Helmert 2010). $-s$ denotes the stack obtained with the reverse sign for every pancakes of s.

Known stacks hard to solve are (Cohen and Blum 1995), (Cibulka 2011): $-I_N$, $J_N = [\mathbf{+1}, -2, ..., -(N-1), -N]$, and $Y_N = [-1, -2, ..., -(N-2), \mathbf{+(N\text{-}1)}, -N]$. We define other stacks $H_{N,M}$ where M is a non-negative integer with N bits. $H_{N,M}(i) = i \times sgn(i, M)$ with $sgn(i, M) = -1 + 2msb(i, M)$. $msb(i, M)$ is the ith most significant bit of M. We have $H_{6,0} = -I_6$, $H_{6,2} = Y_6$, $H_{6,32} = J_6$, and $H_{6,63} = I_6$.

$a(s)$ (respectively $a(-s)$) denotes the number of adjacencies (respectively anti-adjacencies). An adjacency (respectively anti-adjacency) occurs between two neighbouring pancakes when their size difference is 1 and when the burnt side of the smallest (respectively largest) pancake faces the unburnt side of the largest (respectively smallest) pancake (Cibulka 2011).

New Results on $g(-I_N)$

First, we use IDA* with $\#bp$. We use a Linux computer with one core Intel(R) Xeon(R) CPU X5690 running at 3.47GHz. We compute $g(-I_N)$ for N as high as possible. The second leftmost column of Table 1 gives the values of $g(-I_N)$ for $N \leq 27$. Because $\#bp$ is an admissible heuristic, these values are exact. As another result, we have lower bounds of $g(N)$ for $N \leq 27$.

Secondly, we designed a heuristic function h_B: $h_B(s) = \#bp + \lambda a(-s)$. We look for λ such that h_B remains admissible and speeds up the execution as much as possible. The features are computed in $O(1)$. For each $N \leq 15$, $TestSet(N)$ is a set of stacks of size N with their exact values obtained with $\lambda = 0$. $TestSet(N)$ contains positions such that $-I_N$, J_N, Y_N, $H_N,_M$ for specific values of M corresponding to stacks ordered in the unburnt version, but alternating positive pancakes and negative pancakes. $\lambda = 0.44$ is our best value preserving optimality and maximizing speed. $\lambda = 0.44$ enabled our program to uncover new upper bounds of $g(-I_N)$ for $N \leq 30$.

Seeking for Hard Positions

Our process seeking for hard positions starts with $B_2 = \{-I_2\}$. For $i > 2$, for each stack s in B_i, it builds candidate stacks sc by using observation O, it solves them and updates B_{i+1}. Observation O: considering a stack s of size $N-1$, (1) reverse the burnt side of one pancake in s or do

Table 1: Values of $g(-I_N)$ for $N \leq 30$ with running times for $\lambda = 0.44$ and $\lambda = 0$.

N	$g(-I_N)$	h_B	$T(\lambda = 0.44)$	$T(\lambda = 0)$
2	4	2	0.001s	0.002s
3	6	4	0.001s	0.002s
4	8	5	0.002s	0.004s
5	10	7	0.002s	0.004s
6	12	8	0.012s	0.03s
7	14	10	0.15s	0.3s
8	15	11	0.2s	0.5s
9	17	12	2.5s	5s
10	18	14	2.5s	5s
11	19	15	1.5s	5s
12	21	17	14s	1m
13	22	18	10s	40s
14	23	20	4s	15s
15	24	21	17s	30s
16	26	23	1m15s	5m
17	28	24	5m	15m
18	29	25	7m	20m
19	30	27	11m	1h
20	32	28	30m	30m
21	33	30	42m	1h30m
22	35	31	45m	1h30m
23	36	33	6h	14h
24	38	34	6h	15h
25	39	36	9h	28h
26	41	37	10h	20h
27	42	38	1d	2d
28	≤ 44	40	2d	
29	≤ 45	41	5d	
30	≤ 47	43	5d	

Table 2: Values of $g(N)$, d_N, T, and B_N.

N	d_N	T	B_N
2	4	0	$-I_2$
3	6	0	$-I_3$ J_3
4	8	0	$-I_4$ J_4
5	10	0	$-I_5$ J_5
6	12	0	$-I_6$
7	14	0	$-I_7$
8	15	0	$-I_8$
9	17	30s	$-I_9$
10	18	1m	$-I_{10}$
11	19	2m	$-I_{11}$ Y_{11} J_{11}
12	21	4m	$-I_{12}$
13	22	6m	$-I_{13}$ Y_{13} J_{13}
14	23	30m	$-I_{14}$ Y_{14} $H_{14,4}$ J_{14}
15	25	1h	Y_{15} J_{15}
16	26	1h20	$-I_{16}$ $H_{16,4}$ J_{16}
17	28	3h	$-I_{17}$
18	29	5h	$-I_{18}$ Y_{18} $H_{18,4}$ J_{18}
19	30	16h	$-I_{19}$ Y_{19} J_{19} $H_{19,4}$ $H_{19,8}$ $H_{19,10}$ $H_{19,2^{18}+2}$ $H_{19,2^{18}+4}$ $H_{19,2^{18}+8}$
20	32	4d	$-I_{20}$ $H_{20,8}$ J_{20}
21	33	8d	$-I_{21}$ $H_{21,4}$ $H_{21,16}$ $H_{21,18}$ $H_{21,20}$ J_{21} $H_{21,2^{20}+2}$ $H_{21,2^{20}+4}$ $H_{21,2^{20}+16}$ Y_{21}
22	35	>25d	$-I_{22}$ Y_{22} J_{22}

Table 3: EOR variations in N and Level l. L_l are the average lengths. T_i are the average times in seconds.

N	L_0	EOR_0	T_0	L_1	EOR_1	T_1
64	122	1.91	0	97.8	1.53	0.05
128	250	1.95	0	203	1.59	0.62
256	505	1.98	0.01			

N	L_2	EOR_2	T_2	L_3	EOR_3	T_3
8	10.0	1.33	0.01	10.0	1.33	0.02
16	20.3	1.31	0.03	19.9	1.28	2
32	40.7	1.29	1.2			

nothing, (2) add pancake $+N$ or $-N$ at the bottom of s. Table 2 gives the set B_N of hard stacks. d_N is the distance between B_N and I_N. T is the elapsed time. We interrupted the process during iteration 22.

Experimental Optimality Ratios

For the burnt and unburnt pancake problems, 2-approximation algorithms are known (Cohen and Blum 1995), (Fischer and Ginzinger 2005). For the unburnt version, (Bouzy 2015) reaches a 1.04 Experimental Optimality Ratio (EOR) defined as follows. $L_A(p)$ is the length of a solution output of algorithm A on p. Since the optimal length of solutions on p cannot be known for large stacks, $EOR(p) = \frac{L_A(p)}{\#bp}$. Then EOR is the average value over a randomly generated set of stacks. For a calibration purpose, IDA* gives $EOR \geq 1.2$ for $N \leq 15$. Table 3 shows the values of EOR in N for MCS with Cohen and Blum's algorithm.

References

Bouzy, B. 2015. An experimental investigation on the pancake problem. In *IJCAI Computer Game Workshop*.

Cazenave, T. 2009. Nested Monte-Carlo Search. In *IJCAI*, 456–461.

Cibulka, J. 2011. Average number of flips in pancake sorting. *TCS* 412:822–834.

Cohen, D., and Blum, M. 1995. On the problem of sorting burnt pancakes. *DAM* 105–120.

Fischer, J., and Ginzinger, S. 2005. A 2-approximation algorithm for sorting by prefix reversals. In *ESA*, volume 3669 of *LNCS*, 415–425.

Gates, W., and Papadimitriou, C. 1979. Bounds for sorting by prefix reversal. *Discrete Math.* 27:47–57.

Helmert, M. 2010. Landmark heuristics for the pancake problem. In *SoCS*, 109–110.

Korf, R. 1985. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence* 27:97–109.