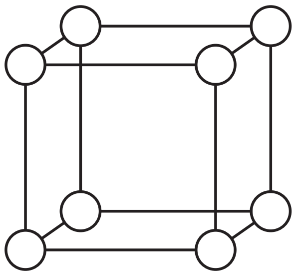
A Heuristic Method of Generating Diameter 3 Graphs for Order/Degree Problem

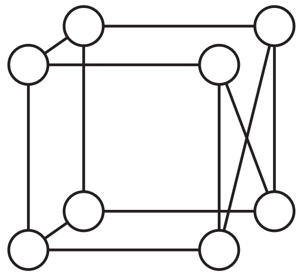
Teruaki Kitasuka and Masahiro Iida

Graduate School of Science and Technology

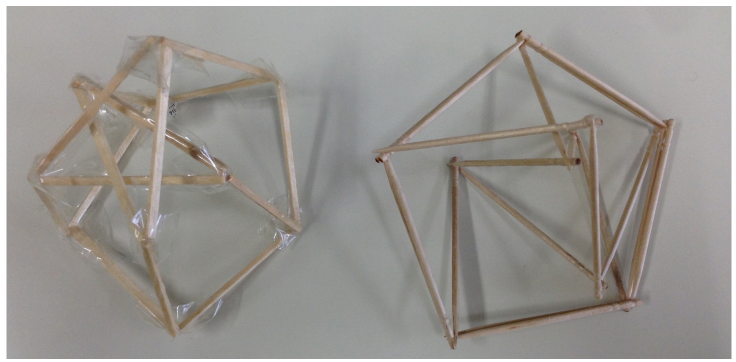
Kumamoto University



Cube; k=3, l=1.71 (Gap 9.1%)



Möbius loop; k=2, l=1.57 (Gap 0.0%)



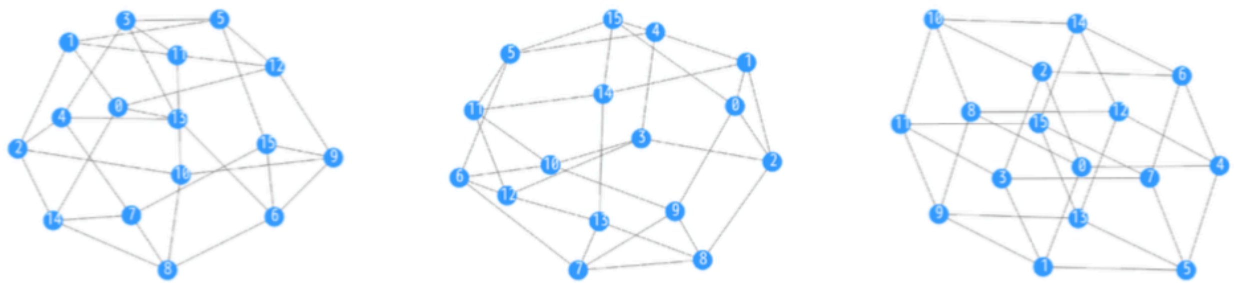
Graph Golf: Order/Degree (OD) Problem

- Find a graph that has smallest diameter & average shortest path length (ASPL) for a given order and degree
  - ✓Find a better topology for various applications
  - ✓Other problem: diameter/degree (DD) problem

● Lessons

Size=16

Degree=4

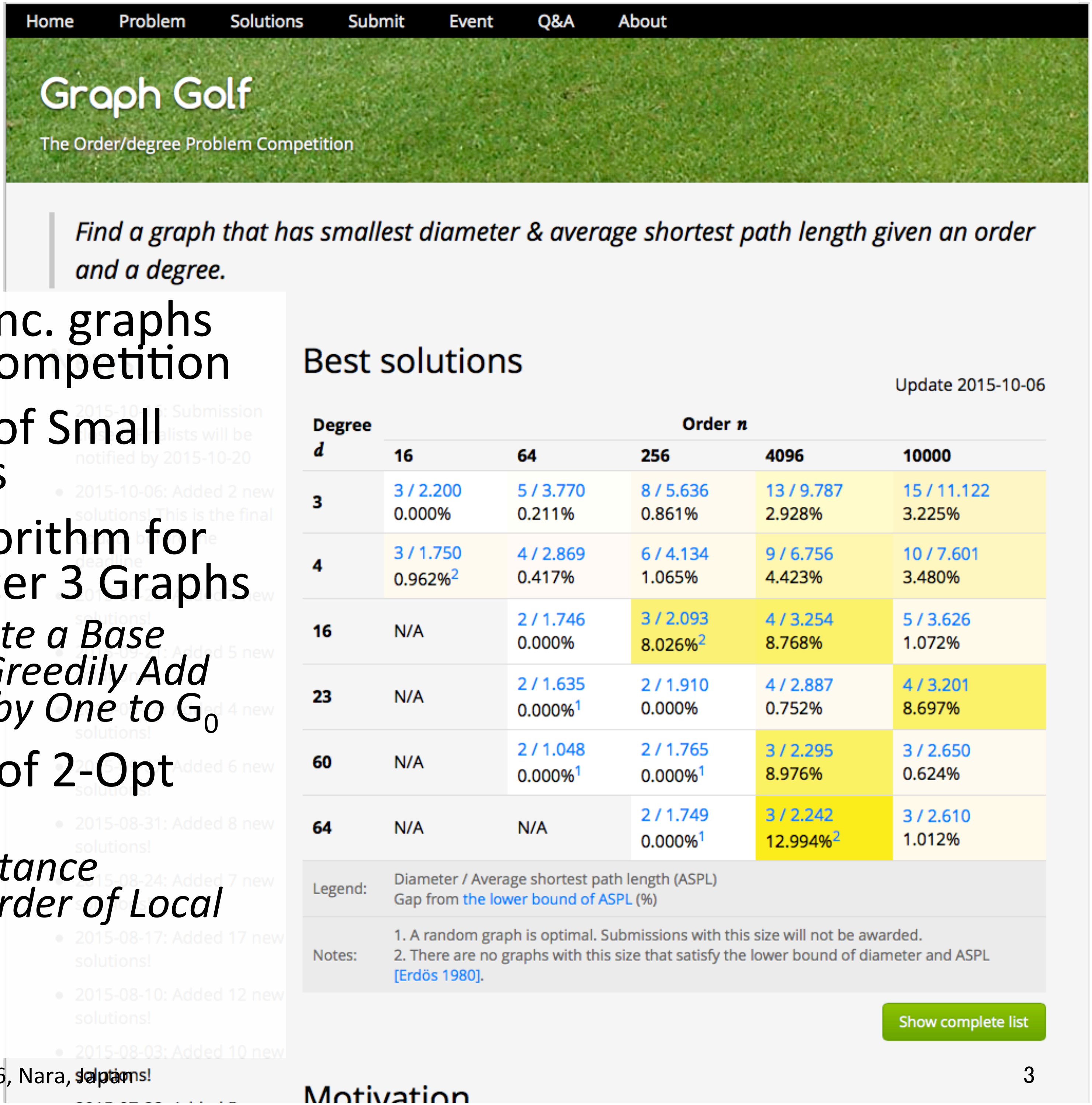


Diameter	3	3	4
Avg. shortest path length	1.942	1.983	2.133

(from The Order/degree Problem Competition Web site http://research.nii.ac.jp/graphgolf/)

Outline

- Our results, inc. graphs found after competition
- Observation of Small Order Graphs
- Heuristic Algorithm for Large Diameter 3 Graphs
  - ✓ *Policy, Create a Base Graph G_0, Greedily Add Edges One by One to G_0*
- A Technique of 2-Opt Local Search
  - ✓ *Edge Importance Function, Order of Local Search*
- Conclusion



Home Problem Solutions Submit Event Q&A About

Graph Golf

The Order/degree Problem Competition

> *Find a graph that has smallest diameter & average shortest path length given an order and a degree.*

Best solutions

Update 2015-10-06

Degree d	Order n: 16	64	256	4096	10000
3	3 / 2.200 0.000%	5 / 3.770 0.211%	8 / 5.636 0.861%	13 / 9.787 2.928%	15 / 11.122 3.225%
4	3 / 1.750 0.962%[2]	4 / 2.869 0.417%	6 / 4.134 1.065%	9 / 6.756 4.423%	10 / 7.601 3.480%
16	N/A	2 / 1.746 0.000%	3 / 2.093 8.026%[2]	4 / 3.254 8.768%	5 / 3.626 1.072%
23	N/A	2 / 1.635 0.000%[1]	2 / 1.910 0.000%	4 / 2.887 0.752%	4 / 3.201 8.697%
60	N/A	2 / 1.048 0.000%[1]	2 / 1.765 0.000%[1]	3 / 2.295 8.976%	3 / 2.650 0.624%
64	N/A	N/A	2 / 1.749 0.000%[1]	3 / 2.242 12.994%[2]	3 / 2.610 1.012%

Legend: Diameter / Average shortest path length (ASPL)
Gap from the lower bound of ASPL (%)

Notes:
1. A random graph is optimal. Submissions with this size will not be awarded.
2. There are no graphs with this size that satisfy the lower bound of diameter and ASPL [Erdös 1980].

Show complete list

Motivation

Our results, including graphs found after competition

Average shortest path length (l)

degree d	Order n: 16	64	256	4096	10000
3	2.200 (July, *2) (= best known)				
4	1.750 (July, *2) (= best known)				
16	(N/A)		2.09069 (After c.., *3) (< 2.09274 best known)	(3.25426873 best known)	(3.62562128 best known)
23					
60				2.295216 (After c., *1) 2.295275 (Aug.)	2.648977 (After c., *3) 2.648980 (After c.) (2.650399 best known)
64				2.242170 (Oct.,, *1) 2.242228 (Sep.,)	2.611310 (After c.) (> 2.610117 best known)

Diameter k (lower bound; best known)

degree d	order n: 16	64	256	4096	10000
3	3; 3	5; 5	7; 8	11; 13	12;15
4	2; 3	4; 4	5; 6	7; 9	8; 10
16	N/A	2; 2	2; 3	4; 4	4; 5
23	N/A	2; 2	2; 2	3; 4	3; 4
60	N/A	2; 2	2; 2	3; 3	3; 3
64	N/A	N/A	2; 2	2; 3	3; 3

Note:
- *1: best solution in comp.
- *2: best but late submission
- *3: best after competition

Observation of Small Order Graphs

Table of the Orders of the Largest Known Graphs for the Degree Diameter Problem

d / *k*	2	3	4	5	6	7	8	9	10
3	10	20	38	70	132	196	336	600	1250
4	15	41	98	364	740	1320	3243	7575	17703
5	24	72	212	624	2772	5516	17030	57840	187056
6	32	111	390	1404	7917	19383	76461	331387	1253615
7	50	168	672	2756	11988	52768	249660	1223050	6007230
8	57	253	1100	5060	39672	131137	734820	4243100	24897161

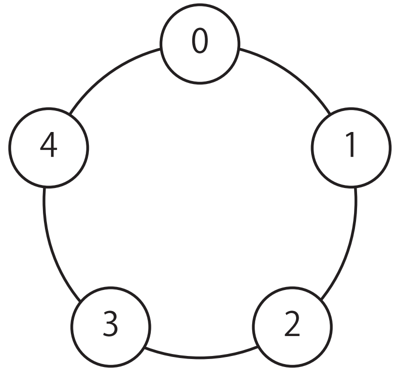
d: degree, *k*: diameter

(pink) The Petersen and Hoffman–Singleton graphs. Moore bound (Optimal)

(green) Other non Moore but optimal graphs

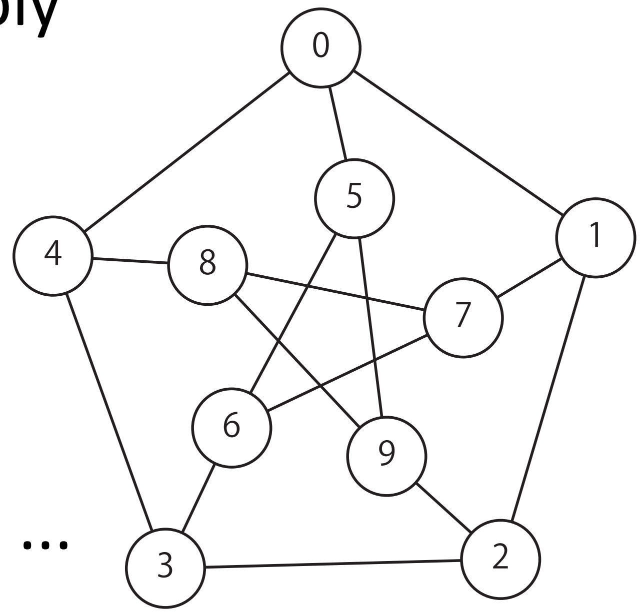
from Combinatorics Wiki
http://moorebound.indstate.edu/wiki/
The_Degree_Diameter_Problem_for_General_Graphs

General Issue of Diameter and Cycles



Diameter *k* = 2 graphs with degree *d*

- *d* = 2: a 5-node cycle
- *d* = 3: Petersen graph (*n* = 10)
- *d* = 7: Hoffman-Singleton graph (*n* = 50)
- *d* > 3: (4 or less)-node cycles are probably redundant



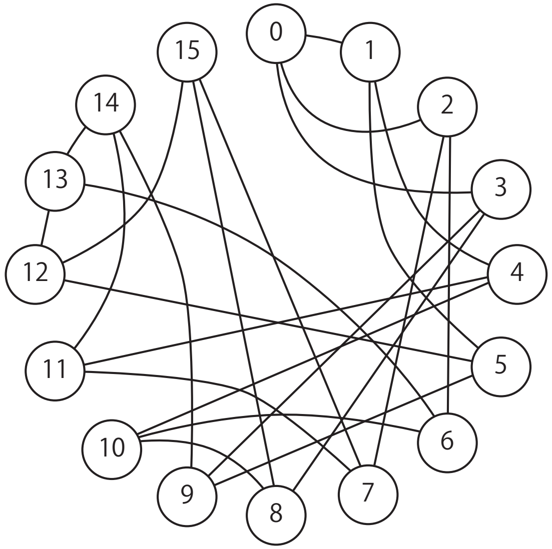
Diameter *k* = 3 graphs with degree *d*

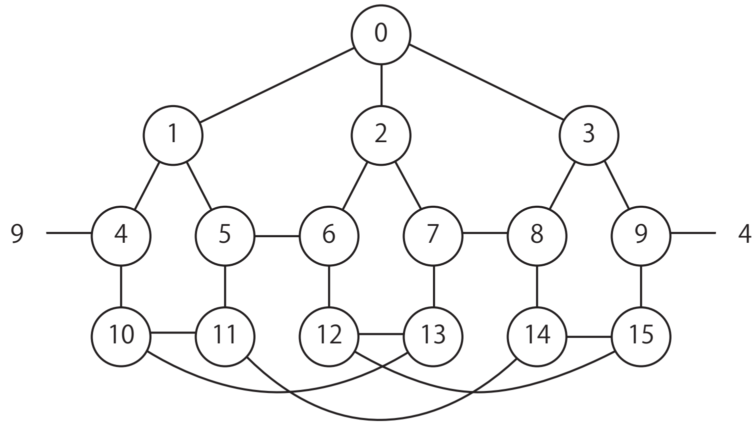
- *d* = 2: 7-node cycle is the best
- *d* > 2: (6 or less)-node cycles are probably redundant

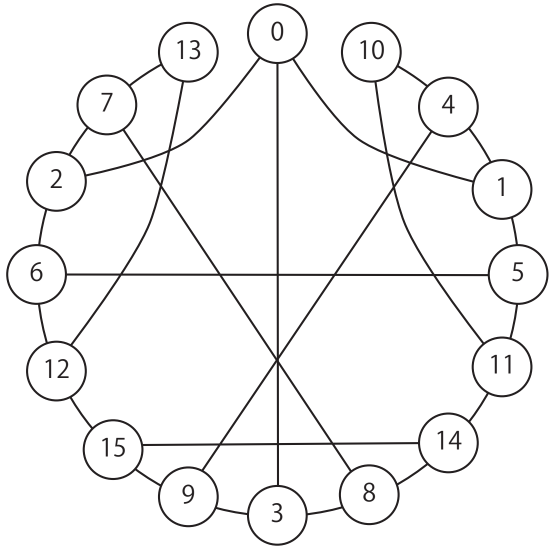
So, try to make 7-node cycles. However, ...

Observation of Small Order Graphs

n = 16, d = 3 graph has k = 3, l = 2.200



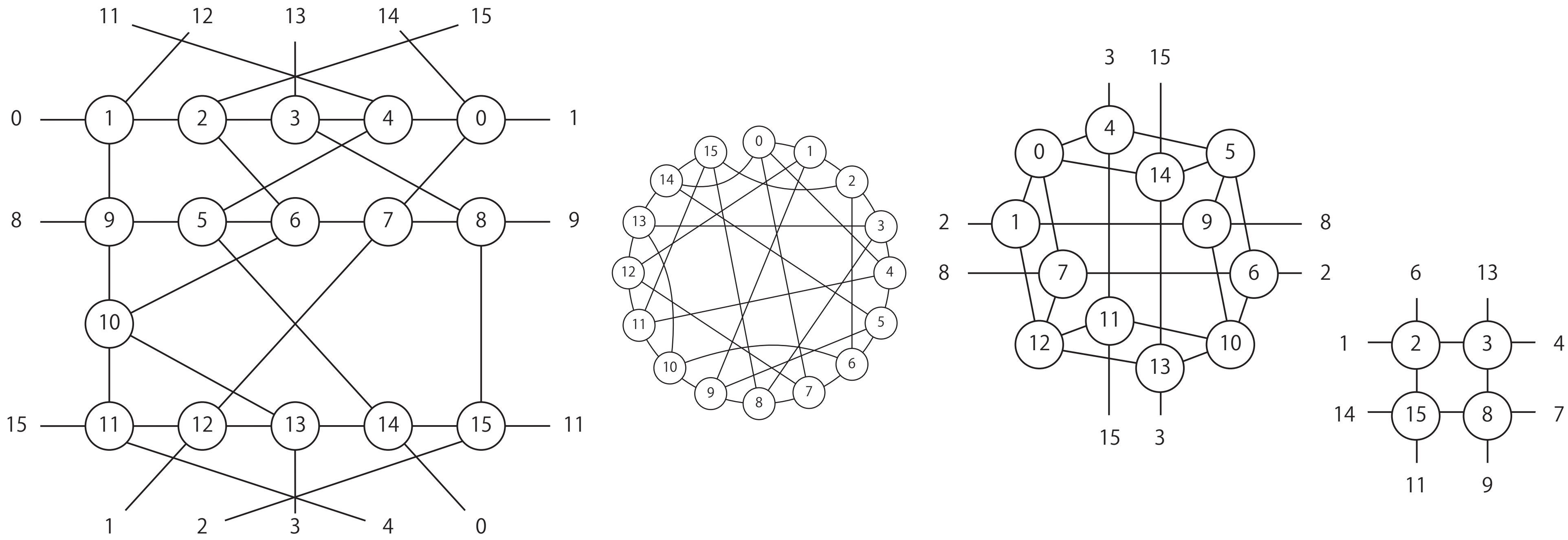




- Many 5-node cycles and no 4-node cycles.
- We will try to increase # of 5−node cycles (pentagons).

Observation of Small Order Graphs

n = 16, d = 4 graph has k = 3, l = 1.750



- Many 5-node cycles.
- # of 4-node cycles is 5 (5-6-10-9, 10-11-12-13, 0-4-5-14, 0-1-12-7, 2-3-8-15)

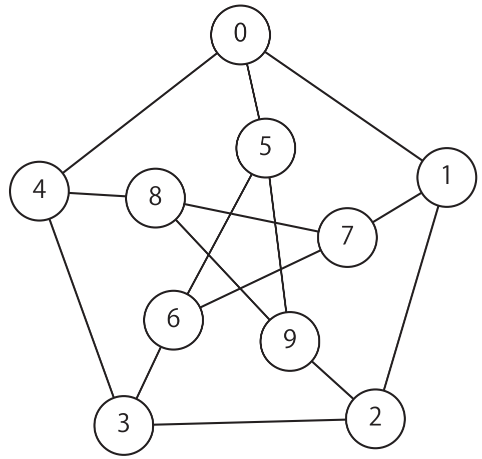
Targeted Graphs of Diameter $k = 3$

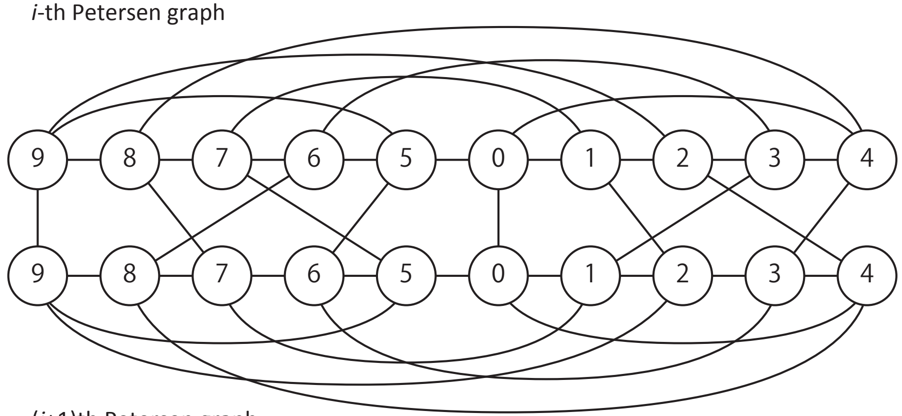
- n = 256, d = 16: k = 3 (Moore bound = 2)
- n = 4096, d = 60 and 64: k = 3 (Moore bound = 3 and 2)
- n = 10000, d = 60 and 64: k = 3 (Moore bound = 3)

degree d	**Order** n		
	256	**4096**	**10000**
16	2.09069 (After c.., *3) 2.12757 (< 2.09274 best known)		
60		2.295216 (After c., *1) 2.295275 (Aug.)	2.648977 (After c., *3) 2.648980 (After c.) (2.650399 best known)
64		2.242170 (Oct.,, *1) 2.242228 (Sep.,)	2.611310 (After c.) (> 2.610117 best known)

Outline of Heuristic Algorithm

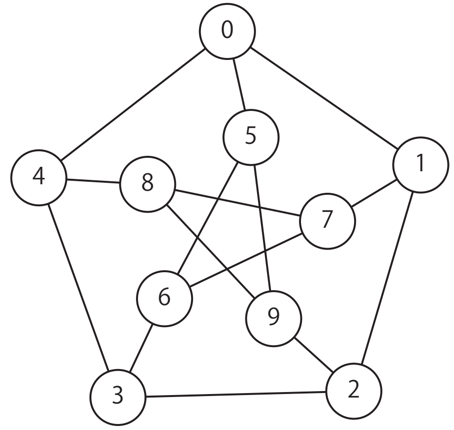
1. Create a Base graph G_0: for targeted diameter k, connect small order graphs of diameter is $k-1$. For $k = 3$, Petersen graph (diameter = 2) is the small graph.
2. Greedily Add Edges One by One to G_0: to increase the number of $(2k-1)$-node cycles (pentagons for $k = 3$).

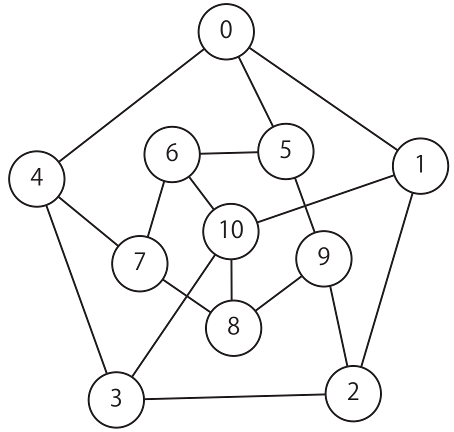




Step 1: Create a Base graph G_0 (1/2)

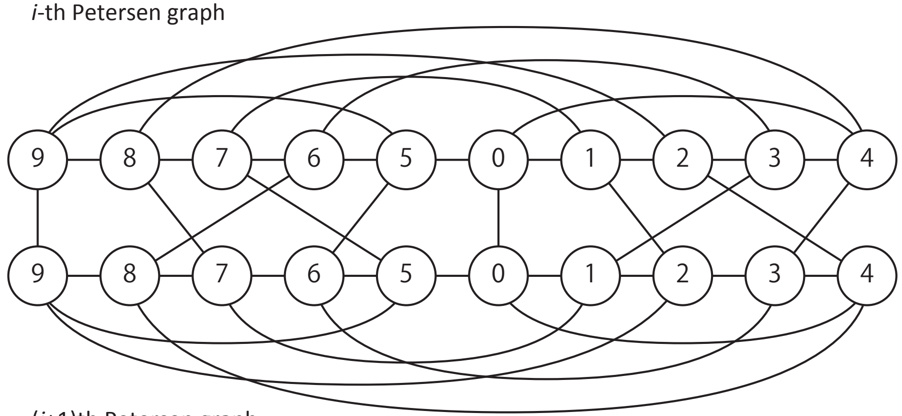
- For targeted diameter k = 3, connect Petersen graphs (diameter is 2).
- n = 10000: 1000 Petersen graphs (diameter = 2) are connected.
- n= 4096: (409 – 6) Petersen graphs and six 11-node graphs are connected.





Step 1: Create a Base graph G_0 (2/2)

- Each small (Petersen) graph is connected with two adjacent graphs.
- When the small graphs are numbered 1, 2, 3, ..., m (m = $\lfloor n/10 \rfloor$), *i*-th and (*i*+1)-th graphs are connected, for *i* = 1, ..., m – 1.
- A node in *i*-th graph is connected with a node in (*i*+1)-th graph.
- Finally, each node has five edges at most.



Step 2: Greedily Add Edges to G_0 (1/2)

Our policies to add an edge:

- Select a node (*i*) which has the smallest degree for one side of new edge.
- Select a node (*j*) for another side of new edge, to increase the number of pentagons.
- No track back.

Algorithm 3 Greedily add edges, one by one to G_0

procedure ADDEDGES(n, d, G_0)
  $G \leftarrow G_0$
  while edge can be added **do**
    Select a node i from the smallest degree nodes
    Compute node set J such that $d(i, j) > 2$
    for each node $j \in J$ **do**
      $p_1(j) = \text{COUNTPATHS}(i, j)$
      $p_2(j) = 0$
      for each $k \in j$'s neighbors **do**
        if $\text{COUNTPATHS}(i, k) > p_2(j)$ **then**
          $p_2(j) = \text{COUNTPATHS}(i, k)$
        end if
      end for
    end for
    Select $j \in J$ that satisfies conditions (1) and (2)
    Add an edge i-j to graph G
  end while
  If degree of several nodes are less than d, add several edges between them
end procedure
function COUNTPATHS(i, j)
  $p = |D_1(i) \cap D_2(j)| + |D_2(i) \cap D_1(j)|$ ▷ Roughly count the number of paths with distance 3 between node i and j, for graph G
end function

Heuristic Algorithm for Large Diameter 3 Graphs

Step 2: Greedily Add Edges to G_0 (2/2)

To increase the number of pentagons.

- CountPaths function roughly counts the number of path of length 3 for a given pair of nodes.
- For each candidate of *j* and each neighbor *k* of *j*, compute CountPaths(*i*, *k*) and select *j* with a *k* which has the largest value of CountPaths.

Algorithm 3 Greedily add edges, one by one to G_0

procedure ADDEDGES(n, d, G_0)
    $G \leftarrow G_0$
    while edge can be added **do**
        Select a node i from the smallest degree nodes
        Compute node set J such that $d(i, j) > 2$
        for each node $j \in J$ **do**
            $p_1(j) = \text{COUNTPATHS}(i, j)$
            $p_2(j) = 0$
            for each $k \in j$'s neighbors **do**
                if $\text{COUNTPATHS}(i, k) > p_2(j)$ **then**
                    $p_2(j) = \text{COUNTPATHS}(i, k)$
                end if
            end for
        end for
        Select $j \in J$ that satisfies conditions (1) and (2)
        Add an edge i-j to graph G
    end while
    If degree of several nodes are less than d, add several edges between them
end procedure
function COUNTPATHS(i, j)
    $p = |D_1(i) \cap D_2(j)| + |D_2(i) \cap D_1(j)|$ ▷ Roughly count the number of paths with distance 3 between node i and j, for graph G
end function

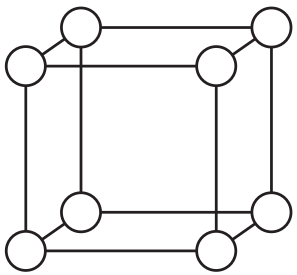
Heuristic Algorithm for Large Diameter 3 Graphs

Targeted Graphs of Diameter $k = 3$

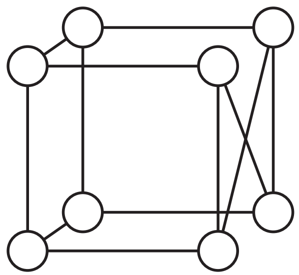
- n = 256, d = 16: k = 3 (Moore bound = 2)
- n = 4096, d = <u>60</u> and <u>64</u>: k = 3 (Moore bound = 3 and 2)
- n = 10000, d = <u>60</u> and 64: k = 3 (Moore bound = 3)

degree d	**Order** n		
	256	**4096**	**10000**
16	2.09069 (After c.., *3) <u>2.12757</u> (< 2.09274 best known)		
60		2.295216 (After c., *1) <u>2.295275</u> (Aug.)	2.648977 (After c., *3) <u>2.648980</u> (After c.) (2.650399 best known)
64		2.242170 (Oct.,, *1) <u>2.242228</u> (Sep.,)	<u>2.611310</u> (After c.) (> 2.610117 best known)

2-Opt Local Search (flip two edges)



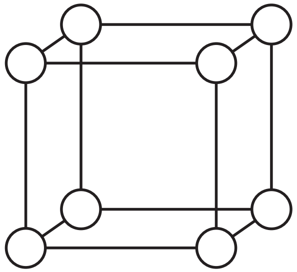
Cube; k=3, l=1.71 (Gap 9.1%)



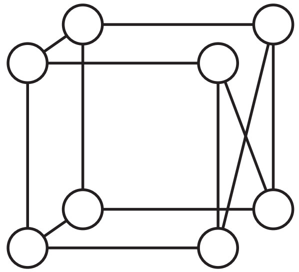
Möbius loop; k=2, l=1.57 (Gap 0.0%)

- 2-Opt: keeping flipped graph, if flipped graph has better diameter / ASPL.
- There are so many pairs of edges.
  # of edges $m = nd/2$. # of pairs of edges = $m(m\text{-}1)/2$.
- For a graph with n = 10,000 and d = 60, $m = 6\times10^5$ and $m(m\text{-}1)/2$ = 179,999,700,000 ≒ 1.8×10^{11}

2-Opt Local Search (flip two edges)



Cube; k=3, l=1.71 (Gap 9.1%)



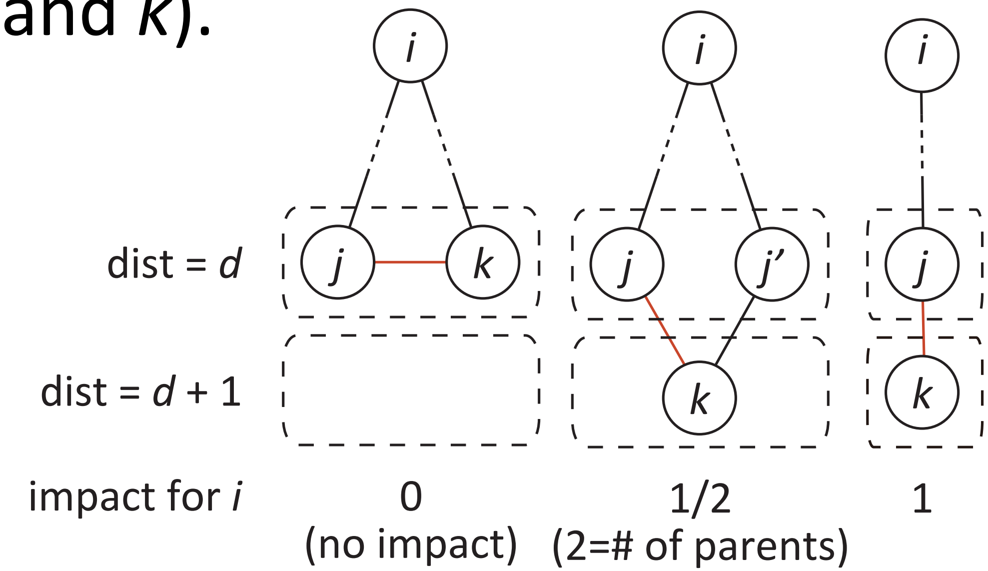
Möbius loop; k=2, l=1.57 (Gap 0.0%)

- Search space of 2-opt local search is very large.
- Question: which pair of edges is a better pair to shorten ASPL?
- An answer: lower important edges are given higher priority

A Technique of 2-Opt Local Search

Edge Importance for fast 2-OPT

- Rough definition: an edge (*j*, *k*) has a positive value for node *i*, if *j* and *k* is <u>not</u> the same distance from *i*. Importance of edge (*j*, *k*) is the sum of such values for all node *i* (*i* ≠ *j* and *k*).

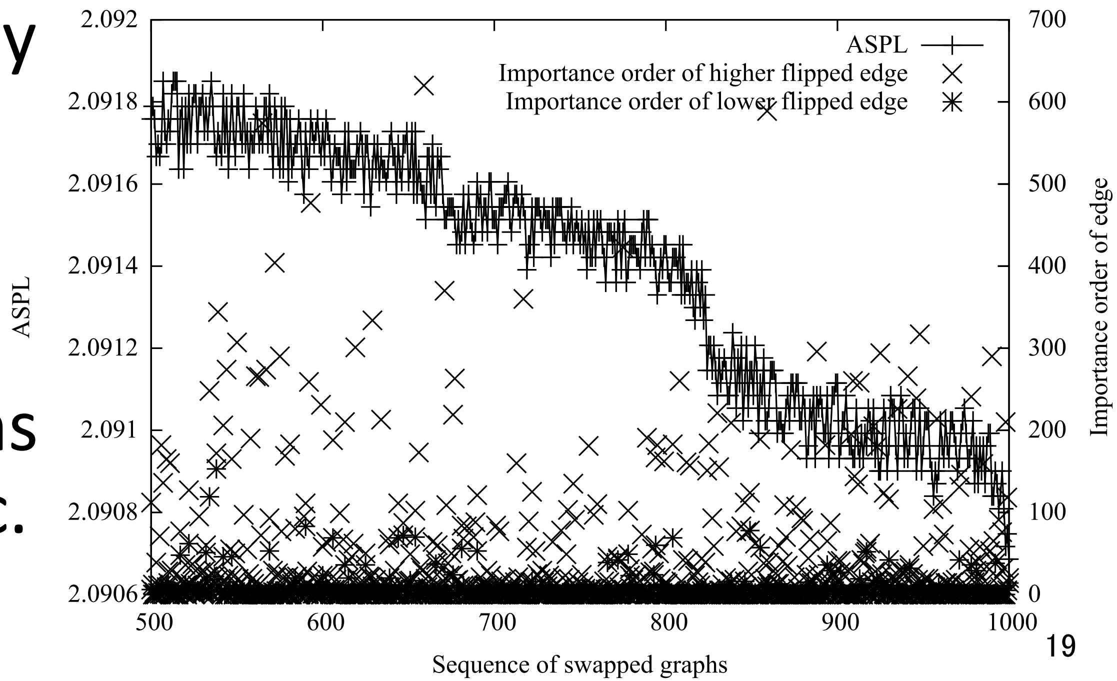


- Calculate importance for all edges, sort them by importance.

A Technique of 2-Opt Local Search

Application Results of Edge Importance

- In a graph of n = 256 and d = 16, there are 2048 edges and 2.1×10^6 edge pairs
- 1000 graphs are searched from l = 2.09258 to 2.09069. In the sequence, we selected edge pair (i, j) from the range of $0 \le i \le 153$ and $1 \le j \le 620$. The range contains only 4.6% of all edge pairs (=154 * 621 / 2.1×10^6)



- Edge importance seems to be a valuable metric.

A Technique of 2-Opt Local Search

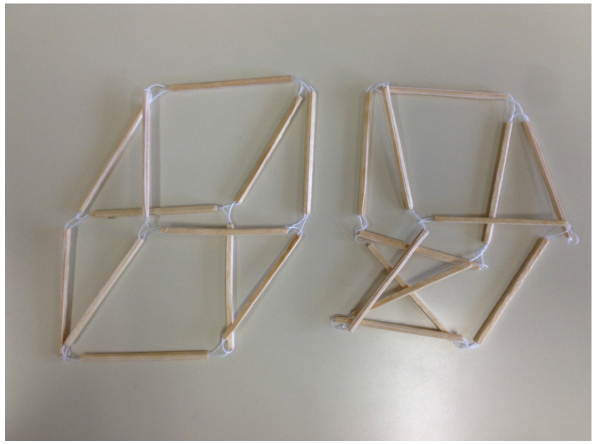
Targeted Graphs of Diameter $k = 3$

- n = 256, d = 16: k = 3 (Moore bound = 2)
- n = 4096, d = 60 and 64: k = 3 (Moore bound = 3 and 2)
- n = 10000, d = 60 and 64: k = 3 (Moore bound = 3)

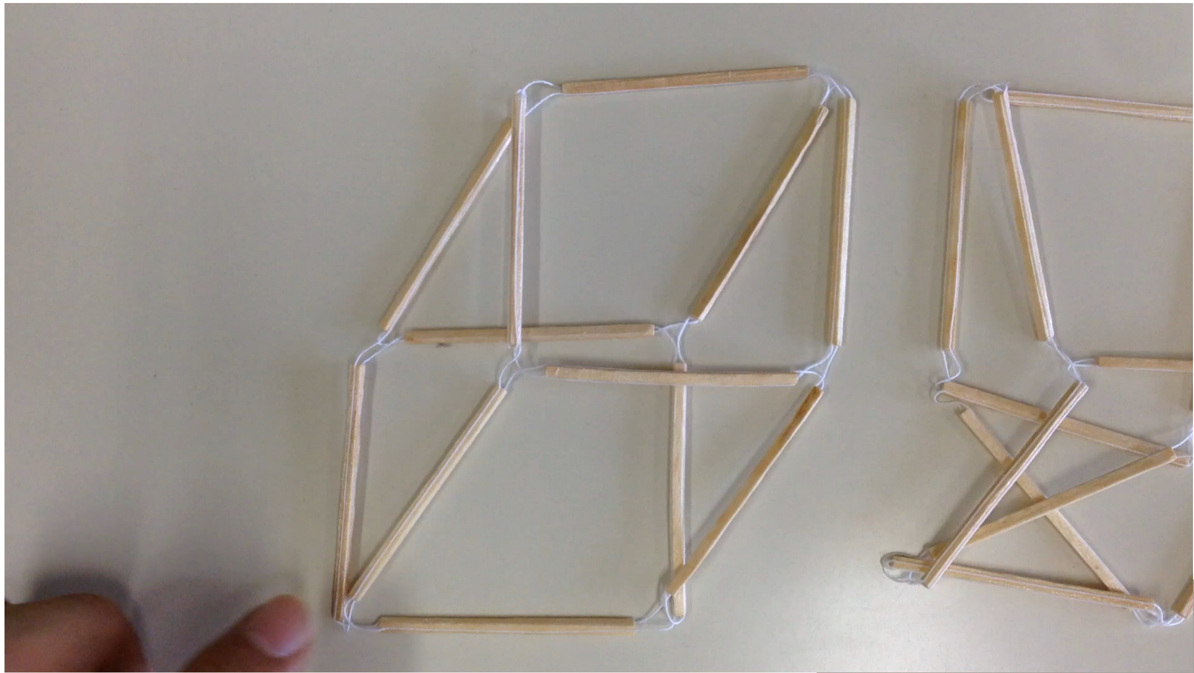
degree d	**Order** n		
	256	**4096**	**10000**
16	2.09069 (After c.., *3) 2.12757 (< 2.09274 best known)		
60		2.295216 (After c., *1) 2.295275 (Aug.)	2.648977 (After c., *3) 2.648980 (After c.) (2.650399 best known)
64		2.242170 (Oct.,, *1) 2.242228 (Sep.,)	2.611310 (After c.) (> 2.610117 best known)

Conclusion

- Explain our results, including graphs found after competition.
- Observing small order graphs, # of pentagon (5-node cycles) is a key property of diameter 3 graphs.
- Explain a heuristic algorithm for large diameter 3 graphs
  - ✓ *Create a Base Graph* G_0, *Greedily Add Edges One by One to* G_0
- Explain a technique of 2-opt local search
  - ✓ *Edge Importance Function, Order of Local Search*
- *Physical handmade graphs (toys)*

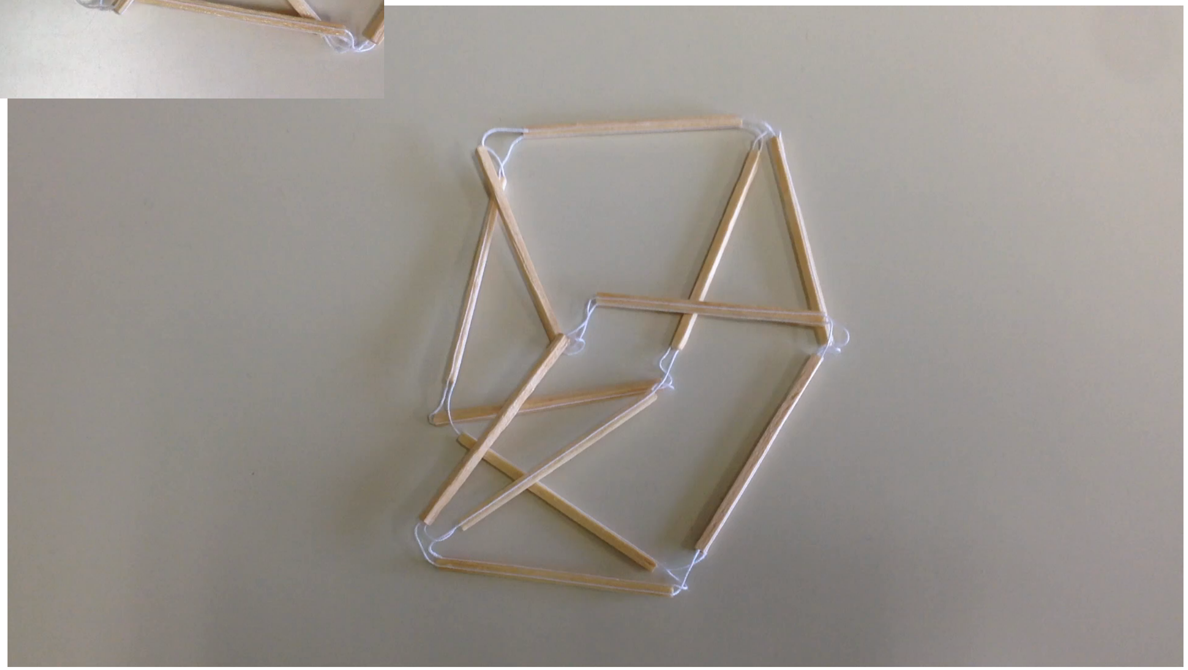


Appendix: Physical Handmade Graphs



Cube

(*k* = 3, *l* = 1.71)



Möbius loop

(k = 2, l = 1.57)