Formulation space search metaheuristic

Nenad Mladenovic

Department of Industrial Engineering, Khalifa University, Abu Dhabi, UAE

Mathematical Institute, SANU, October 5, 2021, Belgrade, Serbia

Outline

1. FSS – Introduction

- Many methods for solving discrete and continuous global optimization problems are based on changing one formulation to another,
- which is either equivalent or very close to it, so that by solving the reformulated
  problem we can easily get the solution of the original one.
- These types of methods include
- (i) dual methods,
- (ii) primal-dual methods,
- (iii) Lagrange methods,
- (iv) linearization methods,
- (v) convexification methods,
- (vi) (nonlinear) coordinate system change methods

    (e.g., polar, Cartesian, projective transformations, etc.),

  (vii)discrete / continuous reformulation methods,

  (viii)augmented methods, to mention a few.

1. FSS – Introduction (cont.)

- In all those classes, the set of formulations of one problem are not considered as a set having some structure provided with some order relation among formulations.
- The usual conclusion in papers oriented to a new formulation is that a given formulation is better than another or the best among several.
- The criteria for making such conclusions are typically the duality or integrality gap provided (difference between upper and lower bounds), precision, efficiency (the CPU times spent by the various methods applied to different formulations of the same instances), and so on.

1. FSS – Introduction (cont.)

- Formulation space search (FSS) is a metaheuristic first proposed in 2005 by Mladenovic, Plastria and Urosevic.
- Since then, many algorithms for solving various optimization problems have been proposed that apply this framework.
- The main idea is to provide the set of formulations used for solving a given class or type of problem with some metric or quasi-metric.
- In that way, the distance between formulations can be induced from those (quasi) metric functions, and thus, the search space is extended to the set of formulations as well.

1. FSS – Introduction (cont.)

- The search space becomes a pair (F, S), consisting of **formulation space** F and **solution space** S.

- **Most importantly**, in all the mentioned method classes (i) - (viii), the discrete metric function between any two formulations can easily be defined.

- For example, in Lagrange methods, the distance between any two formulations can be defined as the difference between their relaxed constraints (the number of multipliers used);

- in coordinate change methods the distance between formulations could be the difference between the number of entities (points) presented in the same coordinate system, etc.

2. Literature review

- **Circle packing** - Reformulation descend (Mladenovic et al. 2005, 2007)
- **Packing unequal circles** (Lopez and Beasley 2013)
- **Packing unequal circles in a fixed size circular container** (L´opez and Beasley, 2016).
- **Mixed Integer nonlinear programming problem** (L´opez and Beasley 2017).
- **Timetabling problem** (Kochetov et al. 2008).

2. Literature review (cont.)

- **Multi-item capacitated lot-sizing** problem (Erromdhani et al. 2017)
- **Graph Colouring** (Hertz et al. 2008)
- **Cut-width minimization** problem (Pardo et al. 2013)
- **Maximum min-sum dispersion** (Amirgaliyeva et al. 2017)
- **Continuos location problems** (Brimberg et al. 2014, 2017).

2. Stochastic FSS

- Let us denote with (φ,x) an incumbent formulation-solution pair, and with fopt=f(φ,x) the current objective function value.

- One can alternate between formulation space F and solution space S in the following ways:

  (i) **Monte-Carlo FSS**. This is the simplest search heuristic through F:

  (a) take formulation - solution pair (φ′,x′)∈(F,S) at random and calculate the corresponding objective function value f′;

  (b) keep the best solution and value;

  (c) repeat previous two steps p (a parameter) times

3.1 Stochastic FSS (cont.)

- .(ii) **Random walk FSS**. For a random walk procedure, we need to introduce neighborhoods of both, the formulation and the solution: $(N(\varphi), N(x)), \varphi \in F$ and $x \in S$.
- Then we simply walk through the formulation-solution space by taking a random solution from such a defined neighborhood in each iteration.
- (iii) **Reduced FSS**. It represents a combination of Monte-Carlo and random walk stochastic search strategies.

3.2 Deterministic FSS

- (i) **Local FSS**. One can perform local search through F as well. That is, find local solution x' for any $\varphi' \in N(\varphi)$ (starting from x as initial solution) and keep the best; repeat this step until there is a formulation in the neighborhood that gives an improvement.

- (ii) **Reformulation descent** (RD). We assume that at least two ($k_{max} \geq 2$) not linearly related formulations of the problem are constructed $(\varphi_k), k = 1, ..., k_{max}$, and that initial solution x and formulation ($k = 1$) are found.

- (iii) **Steepest descent** FSS.

```
1 repeat
2    Using formulation $\varphi_l$ and an optimization code, find a stationary or local optimum point $x'$ starting from $x$;
3    if $x'$ is better than $x$ then
4       $x \leftarrow x'$; $l \leftarrow 1$ (start again from the first formulation) ;
     else
5       $l \leftarrow l + 1$ (change formulation);
     end
  until $l = l_{max}$;
```

Algorithm 1: Reformulation descent

```
repeat
1   Take formulation - solution pair
    $(\varphi, x) \in (F, S)$ at random;
2   if $f(\varphi, x) < f_{opt}$ then
3       $\varphi_{opt} \leftarrow \varphi; x_{opt} \leftarrow x;$
        $f_{opt} \leftarrow f(\varphi, x);$
    end
until stopping condition is met;
```

Algorithm 2: Monte–Carlo FSS

```
repeat
  Take formulation - solution pair ($\varphi'$, $x'$) $\in$ ($\mathcal{F} \cap N(\varphi)$, $\mathcal{S} \cap N(x)$) at random;
  if $f(\varphi', x') < f_{opt}$ then
    $f_{opt} \leftarrow f(\varphi', x')$; $\varphi_{opt} \leftarrow \varphi'$; $x_{opt} \leftarrow x'$;
  end
  Set $\varphi \leftarrow \varphi'$; $x \leftarrow x'$;
until stopping condition is met;
```

Algorithm 3: Random walk FSS

```
1 repeat
    $l \leftarrow 1$;
2   while $l \leq l_{max}$ do
      Take formulation - solution pair
      $(\varphi', x') \in (N_l(\varphi), N(x))$ at random;
3     if $f(\varphi', x') < f_{opt}$ then
        $f_{opt} \leftarrow f(\varphi', x')$, $\varphi \leftarrow \varphi'$; $x \leftarrow x'$;
        $l \leftarrow 1$
      else
        $l \leftarrow l + 1$;
      end
    end
  until stopping condition is met;
```

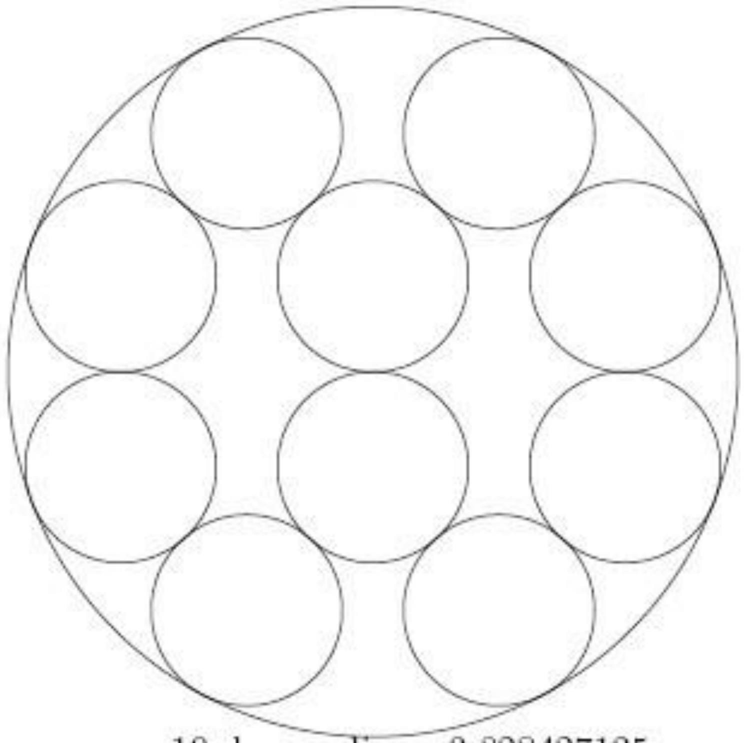
Algorithm 4: Reduced VN FSS

```
function RD-PCC($n$);
$CurrCoord \leftarrow$ Cartesian;
$rcurr \leftarrow$ InitialSolution($n, x, y$);
$rnext \leftarrow$ MinosFull($n, x, y$);
repeat
    $rcurr \leftarrow rnext$;
    if $CurrCoord = Cartesian$ then
        $CurrCord \leftarrow$ Polar;
        CartToPolar($n, x, y, \rho, \alpha$);
        $rnext \leftarrow$ MinosReduced($n, \rho, \alpha$);
    else
        $CurrCoord \leftarrow$ Cartesian;
        PolarToCart($n, \rho, \alpha, x, y$);
        $rnext \leftarrow$ MinosReduced($n, x, y$);
    end
until $rnext \leq rcurr$;
```

Algorithm 5: Reformulation descent for PCC problem

Optimality not so simple.

E.G. $n = 10$

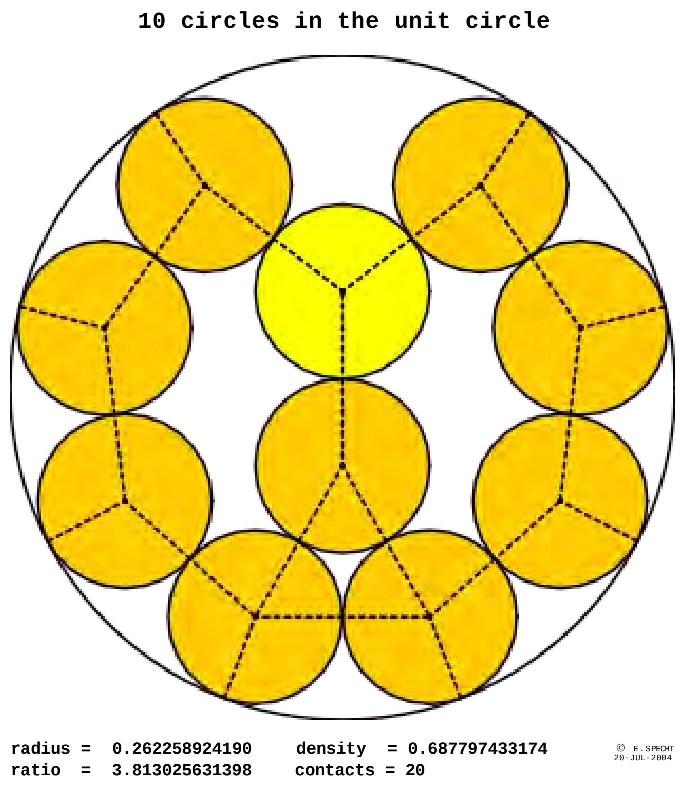


$n = 10$, box radius : 3.828427125

Published by Martin Gardner 1992, following S.Kravitz 1967.

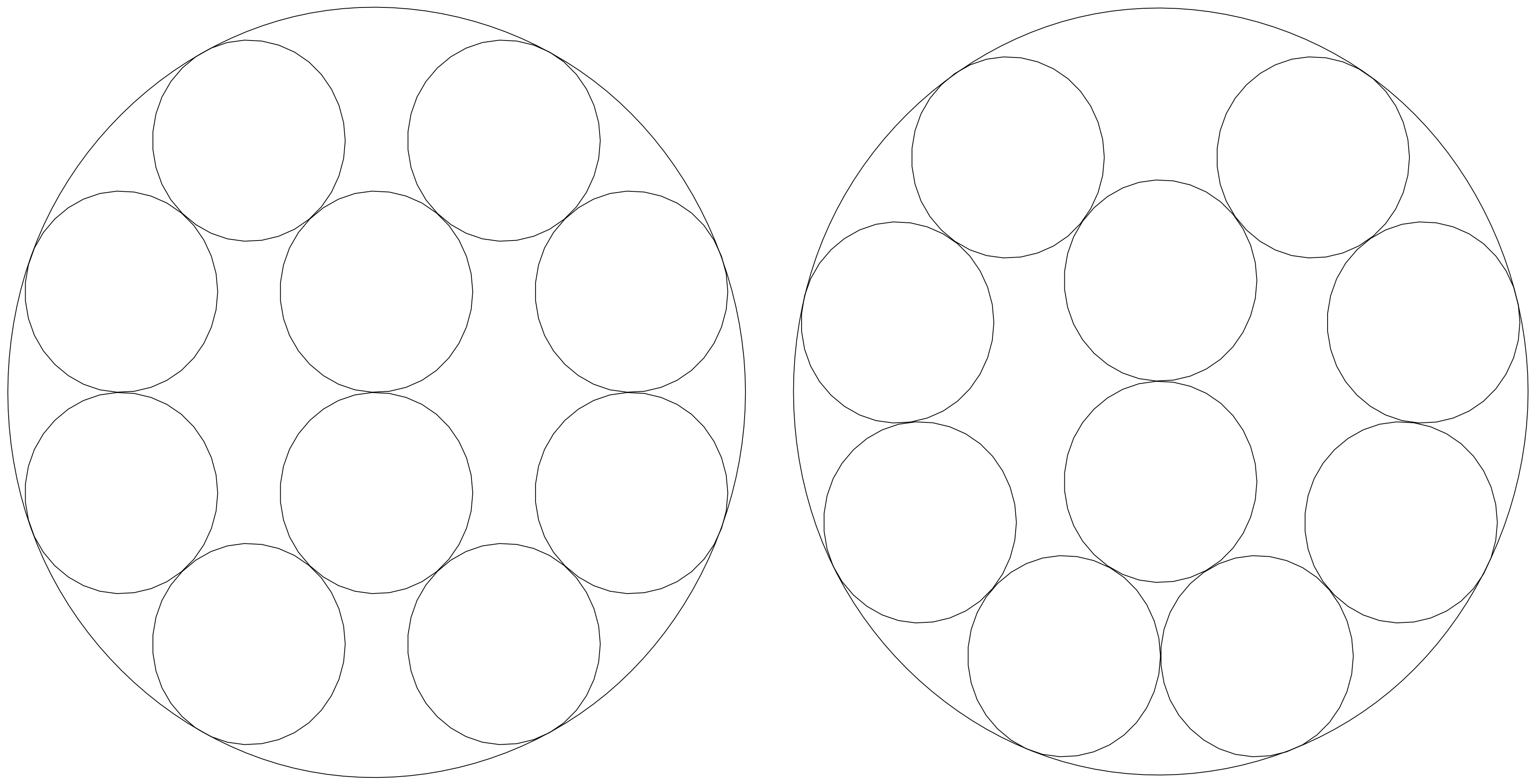
First-order stationary solution of cartesian formulation.

Better solution :



Obtained by a ‘linear’ move in a polar coordinate formulation.

Proven to be optimal by Pirl 1969
according to E. Friedman, website ‘Erich’s Packing Center’
http://www.stetson.edu/~efriedma/packing.html



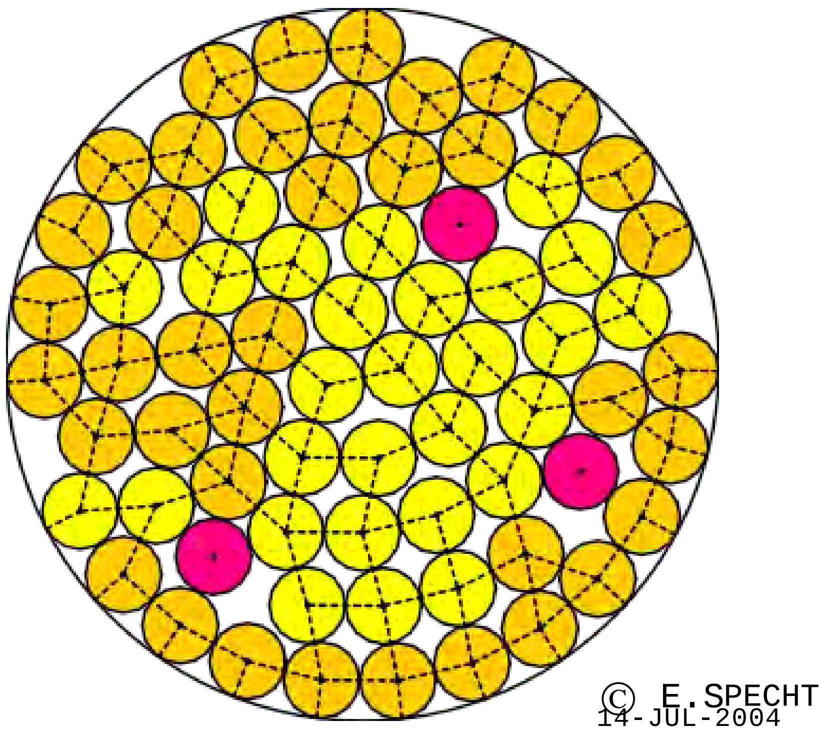
(a) Radius : 3.8284271 (b) Radius : 3.8130256

Figure 1: Packing 10 unit circles into a circle.

Packing circles into a circle

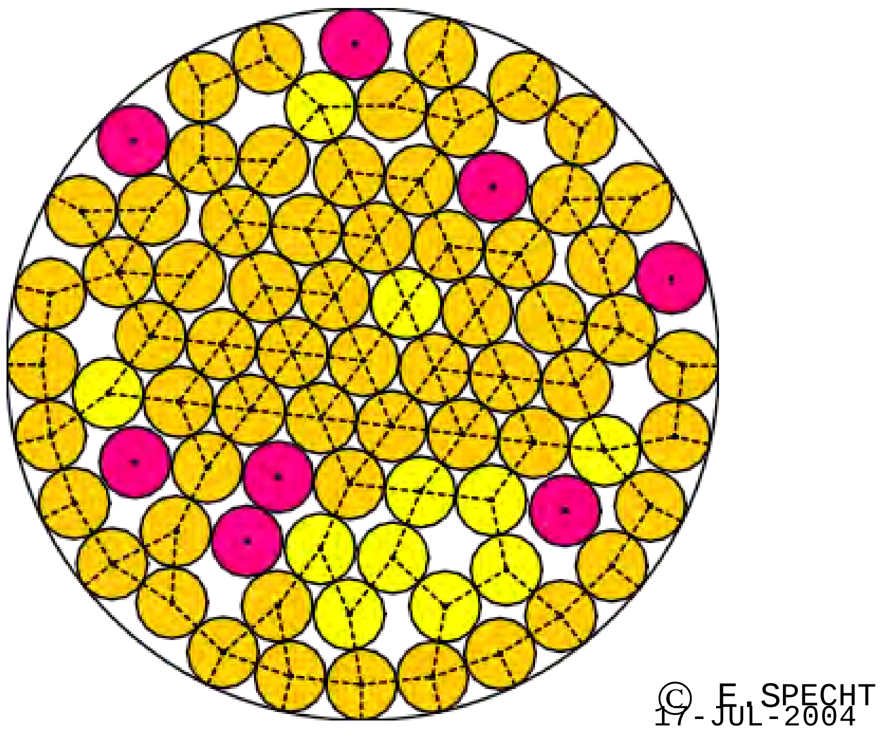
What is the smallest circle in which N unit circles may be packed ?

70 circles in the unit circle



```
radius =  0.106997012559
density  = 0.801385248757
ratio   =  9.346055334486
contacts = 134
```

80 circles in the unit circle



```
radius =  0.100314375962
density = 0.805037921977
ratio   =  9.968660926278
contacts = 146
```

For much more see E.Specht website (updated 25 oct 2008)
http://hydra.nat.uni-magdeburg.de/packing/cci/cci.html

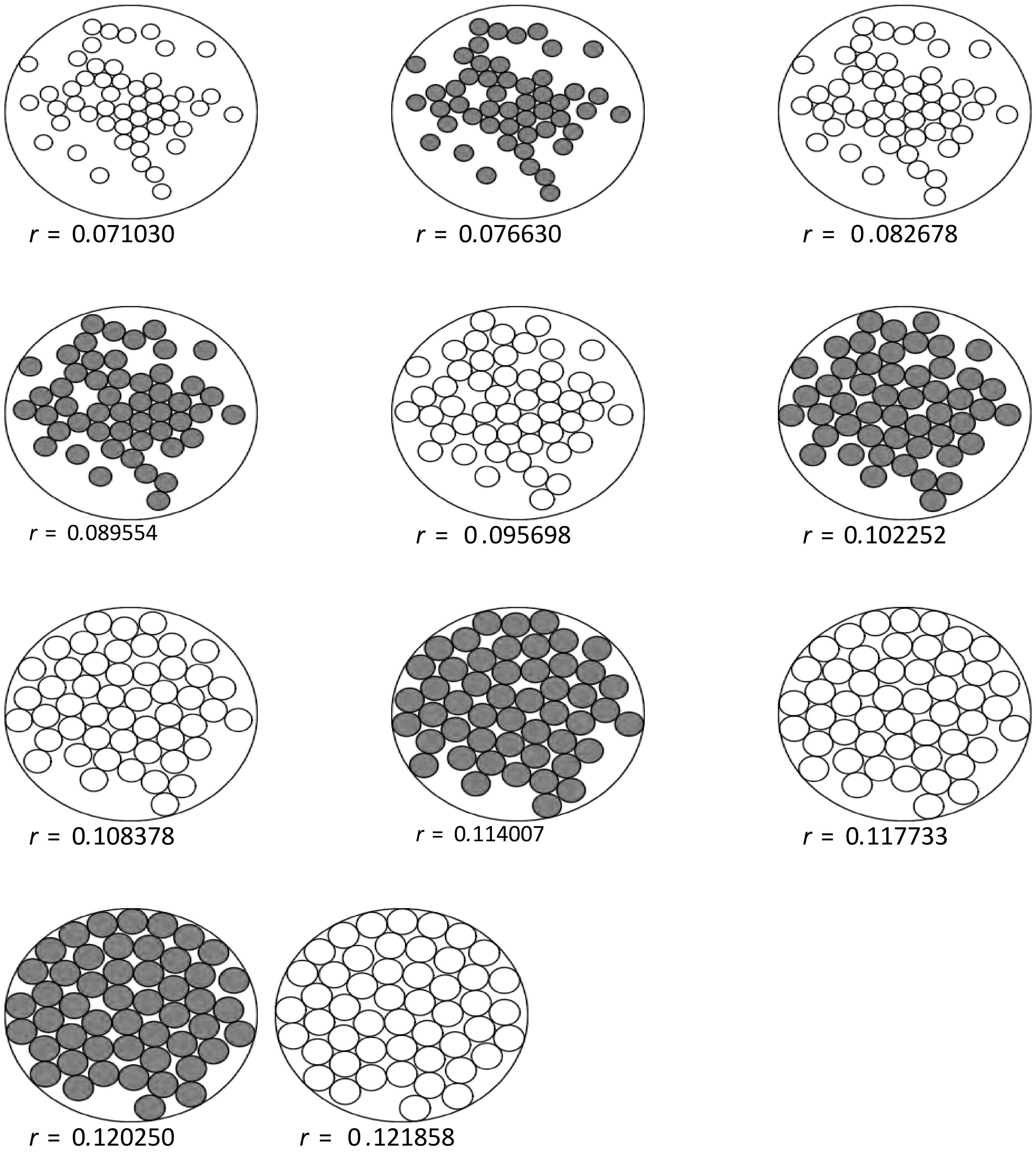


Fig. 1. RD heuristic for PCC and n = 50

8

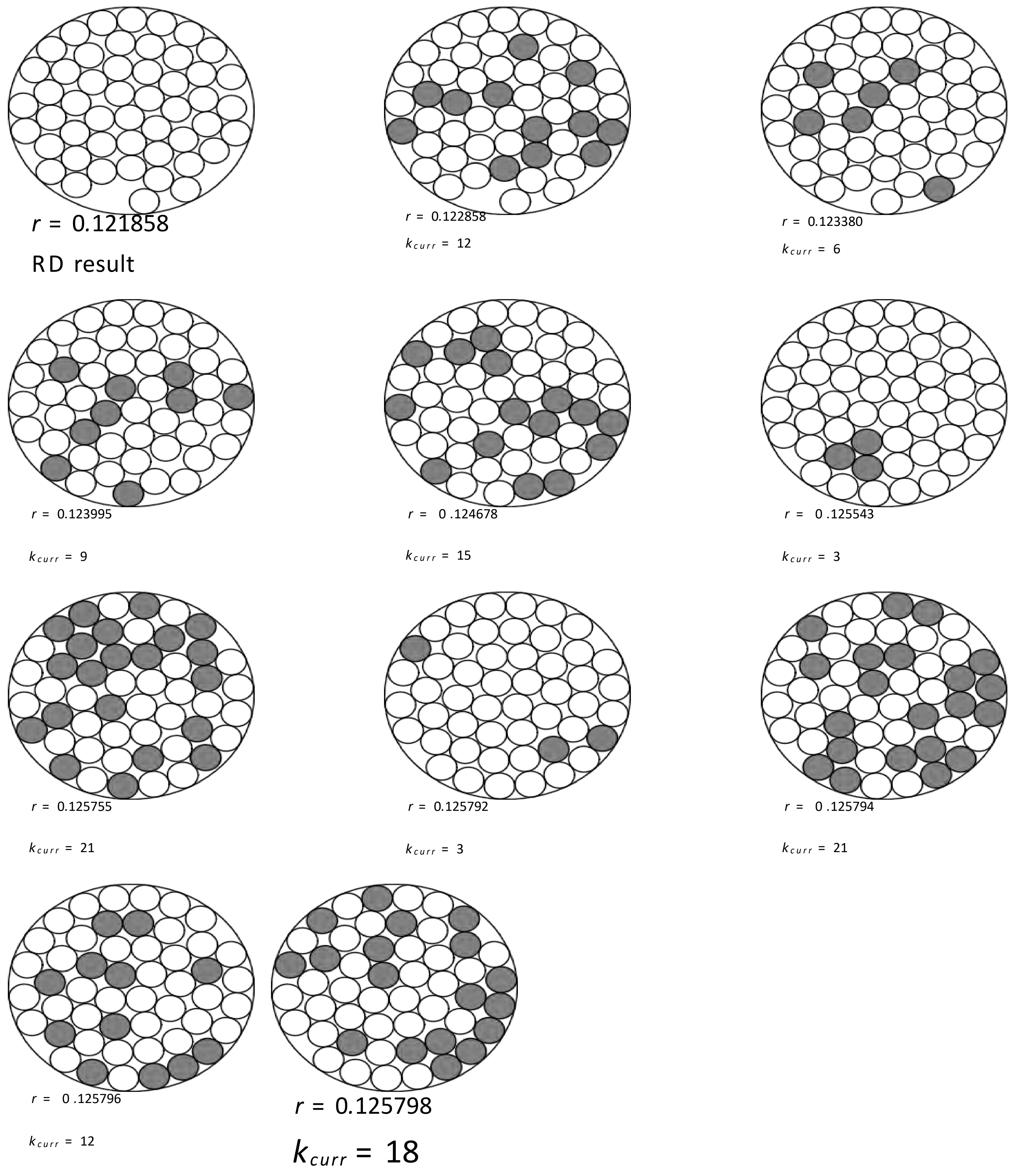


Fig. 2. Reduced VN FSS for PCC problem and *n* = 50.

Formulation Space Search for Circle Packing Problems

Nenad Mladenović[1], Frank Plastria[2], Dragan Urošević[3]
[1]Brunel University, West London, UK, [2]Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussel - Belgium, [3]Matematički Institut SANU, Belgrade, Serbia

Problem

Put given number (n) of circular disks of equal radius within a unit circle without any overlap, and maximize radius of these circles.

Mathematical Formulation

$I = \{1, 2, \dots, n\}$ - labels of disks; Partition $I = C_\varphi \cup P_\varphi$; $C_\varphi \cap P_\varphi = \varnothing$

$$\left\{\begin{array}{ll}
\max r & \\
(x_i - x_j)^2 + (y_i - y_j)^2 - 4r^2 \geq 0 & \forall i, j \in C_\varphi (i \leq j) \\
x_i^2 + y_i^2 \leq (1 - r)^2 & \forall i \in C_\varphi \\
\rho_i^2 + \rho_j^2 - 4\rho_i \rho_j \cos(\alpha_i - \alpha_j) - 4r^2 \geq 0 & \forall i, j \in P_\varphi (i \leq j) \\
\rho_i + r \leq 1 & \forall i \in P_\varphi \\
(x_i - \rho_j \cos(\alpha_j))^2 + (y_i - \rho_j \sin(\alpha_j))^2 - 4r^2 \geq 0 & \forall i \in C_\varphi, \forall j \in P_\varphi \\
r \geq 0 & \\
x_i, y_i \in \mathbb{R} & \forall i \in C_\varphi \\
\rho_i \geq 0, \quad \alpha_i \in [0, 2\pi] & \forall i \in P_\varphi
\end{array}\right.$$

Reduced FSS for PCC problem

```
Function RFSS-PCC($n, kmin, kstep, kmax$);
$rcurr \leftarrow$ RD-PCC($n$);
$rmax \leftarrow rcurr$; $kcurr \leftarrow kmin$;
let $I$ be the set of all centers;
while $Stopping\ Condition\ is\ not\ satisfied$ do
    select subset $P$ of $kcurr$ centers at random; $C = I \setminus P$;
    $rnext \leftarrow$ MinosMixed($n, x, y, \rho, \alpha, C, P$);
    repeat
        $rcurr \leftarrow rnext$; $P = C$; $C = I \setminus P$;
        $rnext \leftarrow$ MinosMixed($n, x, y, \rho, \alpha, C, P$);
    until $rnext \leqslant rcurr$;
    if $rcurr > rmax$ then
        $rmax \leftarrow rcurr$; $kcurr \leftarrow kmin$;
    else
        $kcurr \leftarrow kcurr + kstep$;
        if $kcurr > kmax$ then
            $kcurr \leftarrow kmin$;
        end
    end
end
```

Algorithm 1: Reduced FSS for PCC problem

Possible modification

It is possible to make move in formulation space in the following way:

- Let C_{opt} and P_{opt} be the corresponding sets C and P in last iteration in which we had improvement;
- select subset M of k_{curr} centers at random;
- set $C = (C_{opt} \setminus M) \cup (P \cap M)$ and $P = I \setminus C$

Label this variant with FSS-RC.

Example

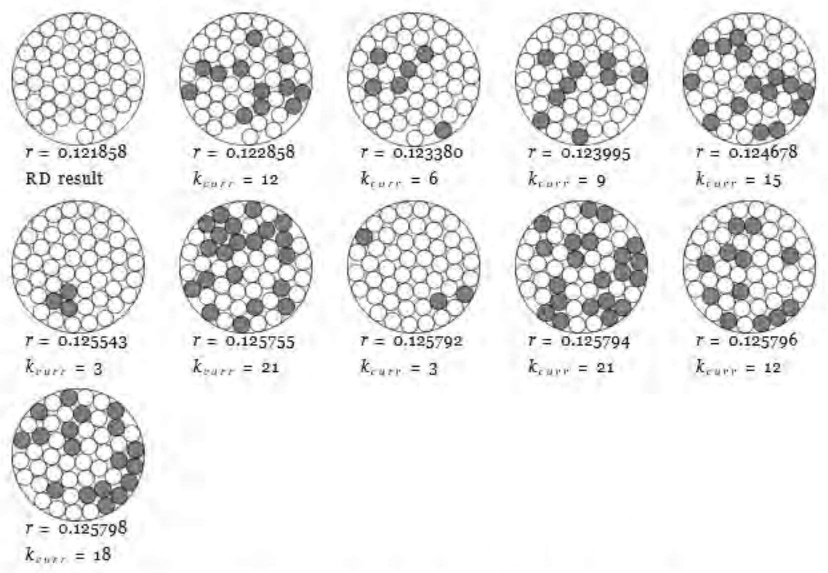


Figure 1: Reduced FSS for PCC problem and n = 50.

Computational results

		RD			FSS			FSS-RC		
n	Best known	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time
50	7.947515	0.06	0.79	3.19	0.00	0.24	80.54	0.00	0.21	72.83
55	8.211102	0.00	2.09	3.37	0.00	0.60	72.81	0.00	0.58	69.17
60	8.646220	0.03	1.40	4.71	0.00	0.95	84.39	0.00	0.89	81.23
65	9.017397	0.00	1.33	16.24	0.00	0.21	108.25	0.00	0.19	96.35
70	9.346660	0.10	0.99	19.56	0.01	0.27	151.64	0.01	0.24	143.18
75	9.678344	0.10	0.77	26.46	0.02	0.20	164.51	0.02	0.18	158.43
80	9.970588	0.10	0.93	39.15	0.04	0.23	229.49	0.03	0.21	206.24
85	10.163112	0.72	1.75	38.79	0.18	0.72	256.17	0.16	0.67	231.49
90	10.546069	0.02	1.27	96.82	0.02	0.56	294.77	0.02	0.53	249.23
95	10.840205	0.18	0.93	147.35	0.07	0.39	308.34	0.05	0.36	281.62
100	11.082528	0.30	1.01	180.32	0.12	0.68	326.67	0.08	0.63	293.18

Mathematical Formulation

$I = \{1, 2, \ldots, n\}$ - labels of disks; Partition $I = C_\varphi \cup P_\varphi$; $C_\varphi \cap P_\varphi = \varnothing$;

$$\max r$$

$$(x_i - x_j)^2 + (y_i - y_j)^2 - 4r^2 \ge 0 \qquad \forall\, i, j \in C_\varphi (i \le j)$$

$$x_i^2 + y_i^2 \le (1 - r)^2 \qquad \forall\, i \in C_\varphi$$

$$\rho_i^2 + \rho_j^2 - 4\rho_i\rho_j \cos(\alpha_i - \alpha_j) - 4r^2 \ge 0 \qquad \forall\, i, j \in P_\varphi (i \le j)$$

$$\rho_i + r \le 1 \qquad \forall\, i \in P_\varphi$$

$$(x - \rho_j \cos(\alpha_j))^2 + (y_i - \rho_j \sin(\alpha_j))^2 - 4r^2 \ge 0 \qquad \forall\, i \in C_\varphi, \forall\, j \in P_\varphi$$

$$r \ge 0$$

$$x_i, y_i \in \mathbb{R} \qquad \forall i \in C_\varphi$$

$$\rho_i \ge 0, \quad \alpha_i \in [0, 2\pi] \qquad \forall\, i \in P_\varphi$$

Computational results

n	Best known	RD			FSS			FSS-RC		
		Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time
50	7.947515	0.06	0.79	3.19	0.00	0.24	80.54	0.00	0.21	72.83
55	8.211102	0.00	2.09	3.37	0.00	0.60	72.81	0.00	0.58	69.17
60	8.646220	0.03	1.40	4.71	0.00	0.95	84.39	0.00	0.89	81.23
65	9.017397	0.00	1.33	16.24	0.00	0.21	108.25	0.00	0.19	96.35
70	9.346660	0.10	0.99	19.56	0.01	0.27	151.64	0.01	0.24	143.18
75	9.678344	0.10	0.77	26.46	0.02	0.20	164.51	0.02	0.18	158.43
80	9.970588	0.10	0.93	39.15	0.04	0.23	229.49	0.03	0.21	206.24
85	10.163112	0.72	1.75	38.79	0.18	0.72	256.17	0.16	0.67	231.49
90	10.546069	0.02	1.27	96.82	0.02	0.56	294.77	0.02	0.53	249.23
95	10.840205	0.18	0.93	147.35	0.07	0.39	308.34	0.05	0.36	281.62
100	11.082528	0.30	1.01	180.32	0.12	0.68	326.67	0.08	0.63	293.18

FSS example: Location-Allocation problem

- The continuous location-allocation problem, also referred to as the multi-source Wet problem, is one of the basic models in location theory.
- The objective is to generate optimal sites in continuous space, notably R^2, for m n facilities in order to minimize a sum of transportation (or service) costs to a set of n fix points or customers with known demands.
- The problem in its most basic form, which will be considered herein, makes the followi assumptions:
  - ▷ there are no interactions between the new facilities;
  - ▷ the number of new facilities (m) is given;
  - ▷ the cost function is a weighted sum of the Euclidean distances between new facilit and fixed points, where the weights are proportional to the flow or interaction betwe the corresponding pairs of locations;
  - ▷ the new facilities have infinite capacities.

LA problem formulation

$$\min_{W,X} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \|x_j - a_i\|$$

$$\begin{aligned} s.t. \quad \sum_{j=1}^{m} w_{ij} &= w_i, \quad i = 1, \ldots, n, \\ w_{ij} &\geq 0, \quad \forall i, j, \end{aligned}$$

- $a_i = (a_{i1}, a_{i2})$ is the known location of customer i, $i = 1, \ldots, n$;
- $X = (x_1, \ldots, x_m)$ denotes the matrix of location decision variables, with $x_j = (x_{j1}, x_{j}$ being the unknown location of facility j, $j = 1, \ldots, m$;
- w_i is the given total demand or flow required by customer i, $i = 1, \ldots, n$;
- $W = (w_{ij})$ denotes the vector of allocation decision variables, where w_{ij} gives the flow customer i from facility j, $i = 1, \ldots, n, \; j = 1, \ldots, m$;
- $\|x_j - a_i\| = [(x_{j1} - a_{i1})^2 + (x_{j2} - a_{i2})^2]^{1/2}$ is the Euclidean norm.

Heuristics for solving LA problem

- Cooper (1964) suggested several heuristics: ALT, p-median,...
- Love and Juel (1983) proposed 4 heuristics (H1-H4)
- Brimberg and Mladenovic (1995) Tabu search
- Brimberg and Mladenovic (1996) VNA
- Hansen, Mladenovic and Taillard (1998) p-median based.
- Brimberg *et al.* (2000) compared almost 20 different heuristics, including several new (G
  VNS, etc.
- Taillard (2002) Fixed neighborhood search based heuristic.
- Salhi, Gamal (2003) - GA
- Zainudin, Salhi (2007) perturbation based heuristic.
- Jabalameli, Ghaderi (2008) Hybrid Memetic and VNS.
- Brimberg, Mladenovic and Salhi (2008) survey.

FSS local search for LA problem

- **Step 1.** Using random initial solution and Cooper's alternate heuristic, find local minimu x_{opt}.
- **Step 2.** Find a set of unoccupied points U, i.e., new facilities from x_{opt} that do not coinci with the current set of fixed points.
- **Step 3.** Add unoccupied facilities obtained in Step 2 to the set of fixed points (n : $n + card(U)$) and solve the related m-Median problem. Denote m-median solution w x_{med}.
- **Step 4.** If $f(x_{med}) = f(x_{opt})$, stop. Otherwise, return to Step 1 with random init solution $= x_{med}$, if $n < 1.4|V|$, or with new random initial solution and $n = |V|$.

	Objective function values			% deviation		CPU Times (sec.)	
m	Old best	VNS	FSS	VNS	FSS	VNS	FSS
5	1851879.88	1851879.88	1851877.25	0.00	0.00	3.78	69.50
10	1249564.75	1249564.75	1249564.75	0.00	0.00	5.87	27.11
15	980132.13	980146.25	980131.69	0.00	0.00	240.05	52.94
20	828802.00	828696.00	828685.69	-0.01	-0.01	32.28	27.92
25	722061.19	722053.56	722000.69	0.00	-0.01	140.16	82.83
30	638263.00	638236.00	638212.50	0.00	-0.01	175.02	135.98
35	577526.63	577523.94	577501.06	0.00	0.00	78.78	90.17
40	529866.19	529755.75	529680.81	-0.02	-0.03	289.02	86.83
45	489650.00	489551.00	489596.78	-0.02	-0.01	264.09	219.00
50	453164.00	453175.88	453149.09	0.00	0.00	137.34	271.00
55	422770.00	422903.06	422647.75	0.03	-0.03	293.13	168.95
60	397784.41	397759.50	397745.47	-0.01	-0.01	70.53	181.17
65	376759.50	376740.81	376630.47	0.00	-0.03	109.39	144.77
70	357385.00	357363.25	357335.34	-0.01	-0.01	277.00	133.50
75	340242.00	340208.19	340168.03	-0.01	-0.02	153.66	97.80
80	326053.19	326107.28	326035.28	0.02	-0.01	279.20	269.98

Table 1: Comparison of VNS and FSS with $t_{max} = 300$ seconds, $k_{max} = p$ for both.

	Objective function values			% deviation		CPU Times (sec.)	
m	Old best	VNS	FSS	VNS	FSS	VNS	FSS
85	313738.19	313769.16	313670.47	0.01	-0.02	289.56	70.36
90	302837.00	302816.50	302565.94	-0.01	-0.09	93.36	234.17
95	292875.09	292782.72	292730.59	-0.03	-0.05	228.70	44.75
100	283113.00	282964.00	282890.41	-0.05	-0.08	220.63	229.91
105	274576.00	274026.78	273705.91	-0.20	-0.32	83.75	298.28
110	265801.00	265329.50	265292.34	-0.18	-0.19	88.13	234.25
115	257605.00	257397.05	257243.98	-0.08	-0.14	200.45	241.50
120	249584.00	249532.33	249409.92	-0.02	-0.07	242.13	267.11
125	242930.00	242599.98	242111.36	-0.14	-0.34	297.59	296.81
130	236154.00	235945.00	235573.94	-0.09	-0.25	271.59	137.39
135	230431.00	229400.88	229169.05	-0.45	-0.55	272.52	154.27
140	224504.00	223530.78	223423.50	-0.43	-0.48	215.14	285.89
145	218279.00	218441.47	217791.58	0.07	-0.22	299.42	299.28
150	212926.00	212404.06	212344.36	-0.25	-0.27	155.97	156.73
Aver.	471575.19	471420.19	471315.09	-0.06	-0.10	183.61	171.88

Table 2: Comparison of VNS and FSS with $t_{max} = 300$ seconds, $k_{max} = p$ for both.

5. Conclusions

- Many methods for solving global optimization problems are based on changing one formulation to another.
- These types of methods include dual, primal-dual, Lagrangian, linearization, surrogation, convexification methods, coordinate system change, discrete/continuous reformulations, to mention a few.
- The main idea of Formulation Space Search (FSS) is to provide the set of formulations for a given problem with some metric or quasi-metric functions.
- In that way, the (quasi) distance between formulations is introduced, and the search space is extended to the set of formulations as well.
- Most importantly, in all solution method classes mentioned, the discrete metric function between any two formulations can easily be defined.

5. Conclusions

- Those simple facts open an avenue to a new approach where heuristics are developed within the FSS framework.
- Instead of a single formulation with corresponding solution space, as in the traditional approach, there are now multiple formulations and solution spaces to explore in a structured way.
- This opens immense possibilities in designing new and powerful heuristics. For example, it may be that new types of distances in the formulation space will make some hard problems easier to solve
- I hope I have convinced the audience that FSS is an exciting direction for future research.

Thank you for your attention

nenadmladenovic12@gmail.com
Nenad.Mladenovic@ku.ac.ae