New Trends in Equilibrium Systems, , 1–16 (2000)

Equal Circles Packing in a Square II. — New Results for up to 100 Circles Using the TAMSASS-PECS Algorithm *

L.G. CASADO AND I. GARCÍA leo@miro.ualm.es, inma@iron.ualm.es
Department of Computer Architecture and Electronics, University of Almería, Spain.

P.G. SZABÓ AND T. CSENDES {pszabo, csendes}@inf.u-szeged.hu
Department of Applied Informatics, József Attila University, Szeged, Hungary

Received 31.7.99

Abstract. In this work a new stochastic optimization algorithm for solving the problem of optimal packing of n non-overlapping equal circles in a square is proposed. It will be shown that our procedure can find most of the optimal solutions for all the problems previously solved and reported in the literature. Results obtained by our algorithm for up to 100 circles are given in numerical and graphical form. For $n = 32, 37, 47, 62$ and 72 the algorithm has obtained better solutions than those reported in the packing literature. In addition, forty new, unpublished packing results are also reported. The found arrangements were validated by interval arithmetic computations.

Keywords: Optimal Packing, Equal Circles, Unit Square.

1. Introduction

The packing circles in a unit square problem has been introduced in detail in the first paper [16]. The version of the problem discussed in the present paper is: locate n points in a unit square, such that the minimum distance m_n between any two points is maximal. The problem of maximizing the minimal pairwise distance of n points, which are contained in a unit square, can be formulated as the following continuous global optimization problem:

$$\begin{array}{c} \mu(x,y) = \min\limits_{1\le i<j\le n} \sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \\ \max \mu(x,y) \\ \hline x, y \in [0,1]^n,\ n > 1 \text{ integer} \end{array} \tag{1}$$

where x_i, y_i are the coordinates of the *i-th* point.

* This work was supported by the SOCRATES-ERASMUS program (25/ERMOB/1998-99), by the Ministry of Education of Spain(CICYT TIC96-1125-C03-03) and by the Consejería de Educación de la Junta de Andalucía (07/FSC/MDM), and by the Grants FKFP 0739/97, OTKA T 016413 and T 017241.

The computational results for the packing circles in a unit square problem, reported in the literature, were obtained using several different procedures. For instance, the method used by R.L. Graham and B.D. Lubachevsky [3] is based on billiards simulation, K.J. Nurmela and P.R.J. Östergård [11] built a minimization of energy function strategy, standard BFGS quasi-Newton algorithm were used by R. Peikert *et al.* [13], nonlinear programming solver (MINOS 5.3) by C.D. Maranas *et al.* in [6] or Cabri-Géomètre software by M. Mollard and C. Payan in [9].

Most of the algorithmic proposals used to solve (1) are based on Global Optimization stochastic approaches, nevertheless several papers ([5], [13]) describing deterministic approaches have been published. Deterministic methods can ensure that their solutions are optimal solutions to (1), and using these kind of methods optimal solutions for $n \leq 27$ have actually been found. The main problem of these deterministic techniques is that as the number of spread points increases, an explosion of the computational burden occurs. For instance, for $n = 7$ the number of times that a non trivial procedure involving a serious amount of computations must be run is 8, but for $n = 14$ is $9,808$ and for $n = 23$ is $288,873,270$. With the best known deterministic algorithm, optimal solutions have been found only for up to $n = 27$. But for these kinds of algorithms is also necessary to a priori know an approximate value or a lower bound of the solution. Thus, stochastic algorithms are useful, at least, to find a good lower bound.

The present paper provides new results to the packing literature for up to one hundred circles. Some of our results represent improvements compared to the best known results; some others confirm results of the packing literature and the rest correspond to unpublished results. All our computational results were obtained by the stochastic algorithm called TAMSASS-PECS (Threshold Accepting Modified Single Stochastic Search for Packing Equal Circles in a Square) described in Section 2. Our numerical results are precise in all digits, and they were verified by interval arithmetic computations. The numerical results and their graphical presentation in Section 3 show to what extent TAMSASS-PECS is able to find optimal or good solutions.

2. The TAMSASS-PECS algorithm

The TAMSASS-PECS algorithm is based on the Threshold Accepting method [2] and on our modified version of SASS (Single Agent Stochastic Search) [4, 7, 8, 14, 15] (MSASS). The TAMSASS-PECS algorithm has specifically been built to solve the packing n equal circles in a square problem, but can be easily extended to pack equal circles in other regular shapes.

The Threshold Accepting framework and the SASS local search algorithm are used for finding a local maximum of a function defined over a search space. As for all stochastic local search algorithms, the probability of finding a global solution grows with the number of trial points. They are based on stepwise improvements of the current solution by movements in the solution space. The following notations and definitions, as in [10], will be used in this paper.

Definition 1. Let $S_F \subseteq S$ be the set of feasible solutions for a maximization problem and S the search space. We require that S is finite. The goal of the *maximization problem* is to:

$$\text{Maximize } f(s) \text{ subject to } s \in S,$$

where $f : S \to I\!R$ is the objective (cost) function and $I\!R$ is the set of real numbers.

Definition 2. A *move* is defined as :

$$d : S(d) \to S$$

where $S(d) \subseteq S$ is the domain of the move. The set of all the moves of the problem is D. We require that the union of the domains for all move sequences in D is the solution set, thus there is no point in the search space that cannot be reached from any point by multiple use of moves.

We denote the application of a move d to a candidate solution $s \in S$ by $d(s)$.

Definition 3. A candidate solution s' is a *neighbour* of the candidate solution s, if

$$s' = d(s) \text{ for some } d \in D$$

The neighbourhood $N(s)$ of a solution $s \in S(d)$ is the union of all the neighbours of s:

$$N(s) = \bigcup_{d \in D} d(s)$$

Definition 4. A candidate solution s is a *global optimum* if there is no other candidate solution s' in the search space S such that $f(s') > f(s)$. A candidate solution s is a *(weak) local optimum*, if there is no other candidate solution s' in the neighbourhood $N(s)$ of s such that $f(s') > f(s)$. If all the neighbours of s have a smaller value of the objective function than $f(s)$, then s is a *strong local optimum*.

Formally, the Threshold Accepting algorithm [2] is very similar to the simulated annealing algorithms. In Threshold Accepting we accept every move that leads to a new solution not much worse than the current solution. To be more specific, if s is the current solution, the proposed next solution $s' \in N(s)$ is accepted as the next solution, if

$$\Delta = f(s) - f(s') < T_h$$

where $T_h > 0$ is the so called threshold level. During the optimization process the threshold level is gradually decreased like the temperature in simulated annealing. In Threshold Accepting, a move which would give a solution much worse than the

Algorithm 1 : The Threshold Accepting Algorithm.

```
Select an initial solution s
while stopping criterion not satisfied do
        while inner loop criterion not satisfied do
                Select s' ∈ N(s)
                Δ = f(s) − f(s')
                if Δ ≤ T_h
                   s = s'
        Decrease threshold T_h
return the best found solution
```

current solution is not accepted, unlike in simulated annealing. The convergence of the Threshold Accepting algorithm has been proved for some cases [1].

The Threshold Accepting method is described in Algorithm 1. The inner loop is performed a constant L times and the stopping criterion is fulfilled when the final value of the threshold level is reached. A new candidate solution can be accepted also when Δ is equal to the threshold level, keeping the sideway moves with the same function value allowed. In [2] a more adaptive inner loop stopping criterion is used: it ends the inner loop after no improvements were made in the last steps.

Random optimization is traditionally based on single agent stochastic search strategies [8]. S.S. Rao [14] and D.C. Karnop [4] used a uniform random variable as the move function, $d(s)$. J. Matyas utilized Gaussian perturbations for the move function with a bias term to direct the search, $\xi \sim N(b, \sigma\mathbf{I})$ [7]. F.J. Solis and J.B. Wets [15] enhanced this approach by evaluating the objective function at $s' = s - \xi s$, if the evaluation at $s' = s + \xi s$ does not improve the current value of the objective function and incorporated a variable perturbation variance. The bias term and this additional function evaluation serve as stochastic equivalents to incorporating momentum and gradient information. The SASS method is given in Algorithm 2.

The variance of the perturbation size, ξ, is controlled by the number of repeated successes and failures, $scnt$ and $fcnt$, respectively, in selecting a neighbour that decreases the value of the objective function. Note that conditions in line 4 are mutually exclusive. The contraction (ct) and expansion (ex) constants, as well as the upper and lower bounds on the standard deviation of the random perturbation ($\sigma_{sup}, \sigma_{inf}$) are set by the user. The standard deviation of the perturbation is increased when it falls below a given lower bound σ_{inf} (line 4). The contraction and expansion of σ are performed when the number of successes, $scnt$, or failures, $fcnt$, are greater than the user given constants, $Scnt$ and $Fcnt$, respectively. The values 0.4 and 0.2, which modify the bias value are retained from F.J. Solis and J.B. Wets's paper [15]. Other values used in [15] are: $ex = 2$, $ct = 0.5$, $Scnt = 5$, $Fcnt = 3$, $\sigma_{sup} = 1.0$ and $\sigma_{inf} = 10^{-5}$. The standard deviation σ specifies the size of the sphere that most likely contains the perturbation vector, and the bias term, b, locates the center of the sphere based on directions of past success. In

Algorithm 2 : The Single Agent Stochastic Search

1 **proc** $SASS(s_0, ex, ct, Scnt, Fcnt, \sigma_{sup}, \sigma_{inf}, NIter)$
2 **var** $b_0 := 0;\ k := 1;\ scnt := 0;\ fcnt := 0;\ \sigma_0 := 1;$
3 **while** $k < NIter$ **do**
4 $$\sigma_k := \begin{cases} ex \cdot \sigma_{k-1} & if\ scnt > Scnt \\ ct \cdot \sigma_{k-1} & if\ fcnt > Fcnt \\ \sigma_{sup} & if\ \sigma_{k-1} < \sigma_{inf} \\ \sigma_{k-1} & otherwise \end{cases}$$
5 Generate a random vector ξ_k with $N(b_k, \sigma_k \mathbf{I})$ distribution;
6 **if** $f(s' = s_k + \xi_k) > f(s_k)$
7 **then**
8 $s_{k+1} := s';$
9 $b_{k+1} := 0.2b_k + 0.4\xi_k;$
10 $scnt := scnt + 1;$
11 $fcnt := 0;$
12 **else**
13 **if** $f(s' = s_k - \xi_k) > f(s_k)$
14 **then**
15 $s_{k+1} := s';$
16 $b_{k+1} := b_k - 0.4\xi_k;$
17 $scnt := scnt + 1;$
18 $fcnt := 0;$
19 **else**
20 $s_{k+1} := s_k;$
21 $b_{k+1} := 0.5b_k;$
22 $fcnt := fcnt + 1;$
23 $scnt := 0;$
24 $k := k + 1;$

line 13, a reverse strategy seeking a better solution in the opposite direction of the original failed move is performed. The stopping criterion is based on the number of iterations.

The TAMSASS-PECS algorithm is based on a Threshold Accepting method and on a modified version of SASS (MSASS), which was specifically designed for improving the current solution s of the packing equal circles in a square problem. The objective function returns the minimum distance from the center of a circle to the other ones, and the problem at hand consists of maximizing this minimum distance for all centers. Thus, the program iteratively does a maximization of the minimum distance between circle centers.

Based on the parameters provided by TAMSASS-PECS, MSASS is in charge of perturbing the location of a point s_i. This perturbation is intended to increase the minimum value of the distances, $(d_{i,j})$, between s_i and any point s_j $(1 \leq j \neq i \leq n)$.

Algorithm 3 TAMSASS-PECS: a threshold accepting algorithm adapted to the packing equal circles in a square problem.

Select an initial solution s
Select an initial value for T_h
Select an initial standard deviation σ
while $\sigma > \sigma_{final}$ **do**
    while *All centers are not visited* **do**
        $s = MSASS(s, \sigma, T_h, NextCenter(s))$
    Decrease T_h
    Decrease standard deviation σ
return *the best found solution*

It moves the point s_i to a new location, s'_i, and computes the value of the new minimum distance $d'_{i,j}$. Following the Threshold Accepting strategy, a move is accepted if $d_{i,j} - d'_{i,j} < d_{i,j}T_h$, where $d_{i,j}T_h$ is the threshold level and $T_h > 0$. New trial locations of point s'_i, are restricted to the neighbourhood of the current location of the point s_i. This neighbourhood is determined by a normal distribution $N(s_i, \sigma\mathbf{I})$ around s_i. While the Threshold Accepting condition is not satisfied, new locations for the point s_i are tested following the classical SASS algorithm, although the number of trials is limited by a given maximum value (see Algorithm 4).

The TAMSASS-PECS algorithm starts with a pseudorandom initial solution (the location of the n points s_i, $1 \leq i \leq n$, which are generated by dividing the square in $t = k \times k$ non-overlapping tiles, where $k = \lceil\sqrt{n}\rceil$, and $\lceil . \rceil$ denotes the smallest integer not less than the argument. Notice that the number of points n is less or equal than the number of tiles t. The first point is located randomly at the center of the first or second tile. The following points are located at the center of a tile which is separated from the previous by one free tile in a row order. The remaining points are randomly allocated at the free tiles, (one point in one tile). Nevertheless, according to our experiences, if the threshold level is very small, then the initial solution is not crucial because large random movements are allowed.

The initial value for the threshold level is $T_h = 0.02$ and the standard deviation σ is equal to the common diameter of the tiles. The TAMSASS-PECS algorithm tries to improve the current solution by an iterative procedure. At every step, the MSASS subroutine is executed for all the n points using the same values for both the standard deviation σ and the threshold level T_h. The criterion to stop this iterative procedure is based on the value of the standard deviation, which is decreased by a factor of 0.99 at every iteration. The threshold level is also decreased by the same factor.

At every iterative step, MSASS is executed for all the points n in an increasing order, which is determined by the value of the minimum distance from any point to the rest of points $(d_{i,j})$. The value of the minimum distance is updated after every execution of MSASS. The description of the TAMSASS-PECS method is given in Algorithms 3 and 4.

Algorithm 4 : The Modified Single Agent Stochastic Search subroutine

```
proc MSASS(s, σ_0, T_h, i)
  var scnt := 0; fcnt := 0; Fcnt := 3; ct := 0.5; σ := σ_0;
    while fcnt < 4 · Fcnt ∧ scnt = 0 do
          σ := { ct · σ   if fcnt > Fcnt
               { σ        otherwise
          Generate a random vector ξ with N(0, σI) distribution;
          s'(i) := s(i) + ξ;
          if f(s(i)) − f(s') ≤ f(s(i))T_h
            then
                s(i) := s'(i);
                scnt := scnt + 1;
            else
                s'(i) := s(i) − ξ;
                if f(s(i)) − f(s') ≤ f(s(i))T_h
                  then
                      s(i) := s'(i);
                      scnt := scnt + 1;
                  else
                      fcnt := fcnt + 1;
          return s
    end
```

3. Computational results

In this section, results obtained by executing the algorithm proposed in Section 2 are compared to the best previously found results reported in the literature. The programs were coded in C and run under Linux on a Pentium II PC with 266 Mhz. Since the underlying problems are very hard, sometimes to obtain the solution of a problem required multiple runs, each of which took CPU time between a few seconds to a few hours. The numerical results in details together with graphical representations are available at
`http://www.inf.u-szeged.hu/~pszabo`.

Tables 1-2 show results of the problem for $2 \leq n \leq 100$. For every value of n, in the second and third columns, the best results reported in the literature and their references are given. The rest of the columns, with headers m_n, c_n, f_n and ρ_n, refer to values obtained by our algorithm, where:

m_n: the solution of the packing problem found by TAMSASS-PECS,

c_n: the number of contacts between circles and between circles and the sides of the square,

f_n: the number of free circles, and

ρ_n: the density values of the packing.

The definitions of c_n, f_n and ρ_n can be found in the previous paper [16]. The last column, (*), indicates whether the results obtained by our algorithm are better, equal or worse than those previously published by other authors. The meaning of symbols used in this column is the following:

+: our result improves all the previously published results,

−: an earlier result is better than our one,

=: the TAMSASS-PECS algorithm has obtained the same result as the best reported one,

N: it is a new result, previously unpublished.

A summary of the results in Tables 1 and 2 are:

- In 20 out of 59 cases the same results as reported in the literature were found. Most of them are optimal solutions of the packing problem.
- In 34 cases our results are worse than those of other authors. Nevertheless, for several cases our algorithm obtained similar final geometrical arrangements, and the difference is only in the accuracy of the obtained numerical values.
- For $n = 32, 37, 47, 62$, and 72 our results improve all of those previously published.
- 40 new results has been obtained. In all these cases, the value of m_n is better than or equal to the new lower bound described in [16].

The accuracy of our results is 10^{-14}, and the feasibility of these solutions was verified by interval arithmetic. According to these tests there are no better packings in the very close neighbourhood of the given results. On the other hand, these interval verification tests cannot prove global optimality. At the end of this section, packing diagrams of the solutions of problem (1) obtained by the TAMSASS-PECS algorithm are shown (see Figures 1–5). Geometrical arrangements have been drawn for $n = 2$ to $n = 100$ and for $n = 150$ just as an example that TAMSASS-PECS is able to obtain a solution which is better than that obtained by the lower bounds described in [16].

4. Summary

A threshold accepting algorithm based on a modified single agent stochastic search procedure was introduced for the solution of the equal circles packing in the unit square problem. As a result of extensive computational efforts, for five specific problems improved packings were found, while in 27 cases the available known packings were confirmed. For forty further problems ($n \leq 100$) first published good packings are reported. Since the underlying algorithm does not make much use of the special structure of the present problem, we expect that it can be well utilised in other packing problems too.

Table 1. Best results of the packing circles in a unit square problem for $2 \le n \le 50$.

n	The best result in the literature	Ref.	Our Results m_n	c_n	f_n	d_n	*
2	1.41421356237309	[12]	1.41421356237309	5	0	0.53901208445264	=
3	1.03527618041008	[12]	1.03527618041008	7	0	0.60964480874135	=
4	1.00000000000000	[12]	1.00000000000000	12	0	0.78539816339744	=
5	0.70710678118654	[12]	0.70710678118654	12	0	0.67376510556580	=
6	0.60092521257733	[12]	0.60092521257733	13	0	0.66395690946413	=
7	0.53589838486224	[12]	0.53589838486224	14	1	0.66931082684079	=
8	0.51763809020504	[12]	0.51763809020504	20	0	0.73096382525390	=
9	0.50000000000000	[12]	0.50000000000000	24	0	0.78539816339744	=
10	0.42127954398390	[3]	0.42127954398390	21	0	0.69003578526417	=
11	0.39820731023684	[12]	0.39820731023684	20	2	0.70074157775610	=
12	0.38873012632302	[12]	0.38873012632301	25	0	0.73846822388404	−
13	0.36609600769643	[12]	0.36609600769623	25	1	0.73326469490355	−
14	0.34891526037401	[12]	0.34891526037401	32	1	0.73567925554268	=
15	0.34108137740210	[12]	0.34108137740210	36	0	0.76205601092668	=
16	0.33333333333333	[12]	0.33333333333333	40	0	0.78539816339744	=
17	0.30615398530033	[12]	0.30615398530033	34	1	0.73355026330232	=
18	0.30046260628866	[12]	0.30046260628866	38	0	0.75465335787566	=
19	0.28954199199498	[12]	0.28954199199356	37	2	0.75230789673638	−
20	0.28661165235168	[12]	0.28661165235168	44	0	0.77949368686760	=
21	0.27181225535931	[3]	0.27181168718966	40	2	0.75335522651101	−
22	0.26795840155072	[3]	0.26795835833157	42	1	0.77167991577529	−
23	0.25881904510252	[3]	0.25881904510252	56	0	0.76363103212612	=
24	0.25433309503024	[3]	0.25433309503024	56	0	0.77496325975782	=
25	0.25000000000000	[3]	0.25000000000000	60	0	0.78539816339744	=
26	0.23873475724122	[3]	0.23873475724121	56	2	0.75846909048393	−
27	0.23584952830141	[3]	0.23584952830050	55	0	0.77231145646250	−
28	0.23053549364267	[3]	0.23053540627071	56	1	0.77185363595323	−
29	0.22688290074421	[3]	0.22688290074420	65	1	0.77890624177970	−
30	0.22450296453109	[3]	0.22450296453108	65	0	0.79201902646073	−
31	0.21754729161912	[3]	0.21754726920445	54	4	0.77729734717346	−
32	0.21308235294443	[3]	0.21317456258979	63	3	0.77600412447400	+
33	0.21132838414326	[3]	0.21132833175032	64	1	0.78885198102597	−
34	0.20560464675957	[3]	0.20560464675956	80	0	0.77664906433227	−
35	0.20276360086323	[3]	0.20276360086322	80	0	0.78122721299871	−
36	0.20000000000000	[3]	0.20000000000000	84	0	0.78539816339744	=
37	0.19623810145141	[3]	0.19642918368533	73	2	0.78330271742989	+
38	0.19534230412691	[3]	0.19534202418353	74	0	0.79704064569288	−
39	0.19436506316151	[3]	0.19436506314440	80	0	0.81117902717206	−
40	0.18817552201832	[3]	0.18817551865770	85	2	0.78797949519086	−
41	0.18609951184812	[3]	0.18609948922999	97	1	0.79272373684650	−
42	0.18427707211710	[3]	0.18427707211709	90	0	0.79868427865342	−
43	0.18019113545743	[3]	0.18019112440214	83	1	0.78726408457444	−
44	0.17863924567120	[3]	0.17863916198187	79	4	0.79384217677293	−
45	0.17571631417559	[3]	0.17571631417546	94	3	0.78944426849399	−
46	0.17445936087241	[3]	0.17445933585066	90	1	0.79718693728810	−
47	0.17126830721141	[3]	0.17127055746101	93	2	0.78929383201248	+
48	0.16938210954876	[3]	0.16938050793544	97	1	0.79094499029400	−
49	0.16738607686833	[3]	0.16738607686832	120	1	0.79121698952690	−
50	0.16645462588286	[3]	0.16645461022968	102	0	0.79967929970534	−

Table 2. Best results of the packing circles in a unit square problem for $51 \leq n \leq 100$.

n	The best result in the literature	Ref.	Our Results m_n	c_n	f_n	d_n	*
51	0.16561837431260	[3]	0.16561390218764	90	2	0.80861948195609	–
52	0.16538623796964	[3]	0.16538621483322	103	0	0.82253064459417	–
53	–	–	0.16264607633837	93	4	0.81462525898420	N
54	0.15913951630719	[3]	0.15913951414578	109	2	0.79940760435030	–
55	0.15755574752972	[3]	0.15755573439859	105	2	0.80027118200754	–
56	0.15615650046215	[3]	0.15615650046214	119	0	0.80235172950297	–
57	–	–	0.15474723074928	107	3	0.80396408840952	N
58	–	–	0.15130206629471	109	6	0.78673600284850	N
59	–	–	0.15029765718520	118	3	0.79108990822535	N
60	0.14950565404867	[3]	0.14945461308474	125	3	0.79666571193460	–
61	0.14854412669518	[3]	0.14854408720568	119	1	0.80137375361858	–
62	0.14569394327531	[3]	0.14671467098564	114	5	0.79710938267796	+
63	–	–	0.14667828199655	118	2	0.80961563357266	N
64	–	–	0.14532088220301	132	3	0.80922920583758	N
65	–	–	0.14456113665810	112	5	0.81438235586667	N
66	–	–	0.14380319561470	120	2	0.81934777745264	N
67	–	–	0.14304397643538	118	5	0.82409628091674	N
68	–	–	0.14288350731175	115	6	0.83475507341762	N
69	–	–	0.13993726983993	117	3	0.81666484211077	N
70	–	–	0.13787246738894	114	5	0.80715295095497	N
71	-	–	0.13593878298879	133	3	0.79859232191938	N
72	0.13549029317569	[3]	0.13569567607132	129	7	0.80729163319538	+
73	–	–	0.13389894938934	135	5	0.79949982777920	N
74	–	–	0.13265445602583	141	6	0.79720572964583	N
75	–	–	0.13241101073542	108	11	0.80536208356566	N
76	–	–	0.13094762506744	127	9	0.80022799448052	N
77	–	–	0.13047875975659	120	6	0.80562963362431	N
78	0.13046077259640	[3]	0.13045661660359	157	3	0.81584734690826	–
79	–	–	0.12976111574313	131	8	0.81852673455691	N
80	–	–	0.12886897358290	147	5	0.81882205826177	N
81	–	–	0.12833078028621	170	3	0.82293152331831	N
82	–	–	0.12646140351791	145	4	0.81168420885740	N
83	–	–	0.12609559154647	167	3	0.81736730901968	N
84	–	–	0.12555071002762	152	5	0.82087566500715	N
85	–	–	0.12511540460076	146	4	0.82553641420847	N
86	–	–	0.12500381019147	123	7	0.83392471939050	N
87	–	–	0.12230707667418	158	7	0.81150070836153	N
88	–	–	0.12099893725078	162	10	0.80523985787757	N
89	–	–	0.12080111590653	140	14	0.81201615746621	N
90	–	–	0.11988928132785	160	7	0.81010801424488	N
91	–	–	0.11916044953221	172	8	0.81023468623028	N
92	–	–	0.11854559378259	175	4	0.81159835604241	N
93	–	–	0.11827868195489	167	6	0.81711971813354	N
94	–	–	0.11733029717908	158	10	0.81409469757729	N
95	–	–	0.11668436071663	174	4	0.81466286707370	N
96	–	–	0.11633897630581	192	5	0.81887839722130	N
97	–	–	0.11618529717218	179	4	0.82545113090290	N
98	–	–	0.11610339843903	170	7	0.83290785939723	N
99	–	–	0.11601228667056	193	2	0.84022403037368	N
100	–	–	0.11456309641300	184	6	0.82979353948491	N

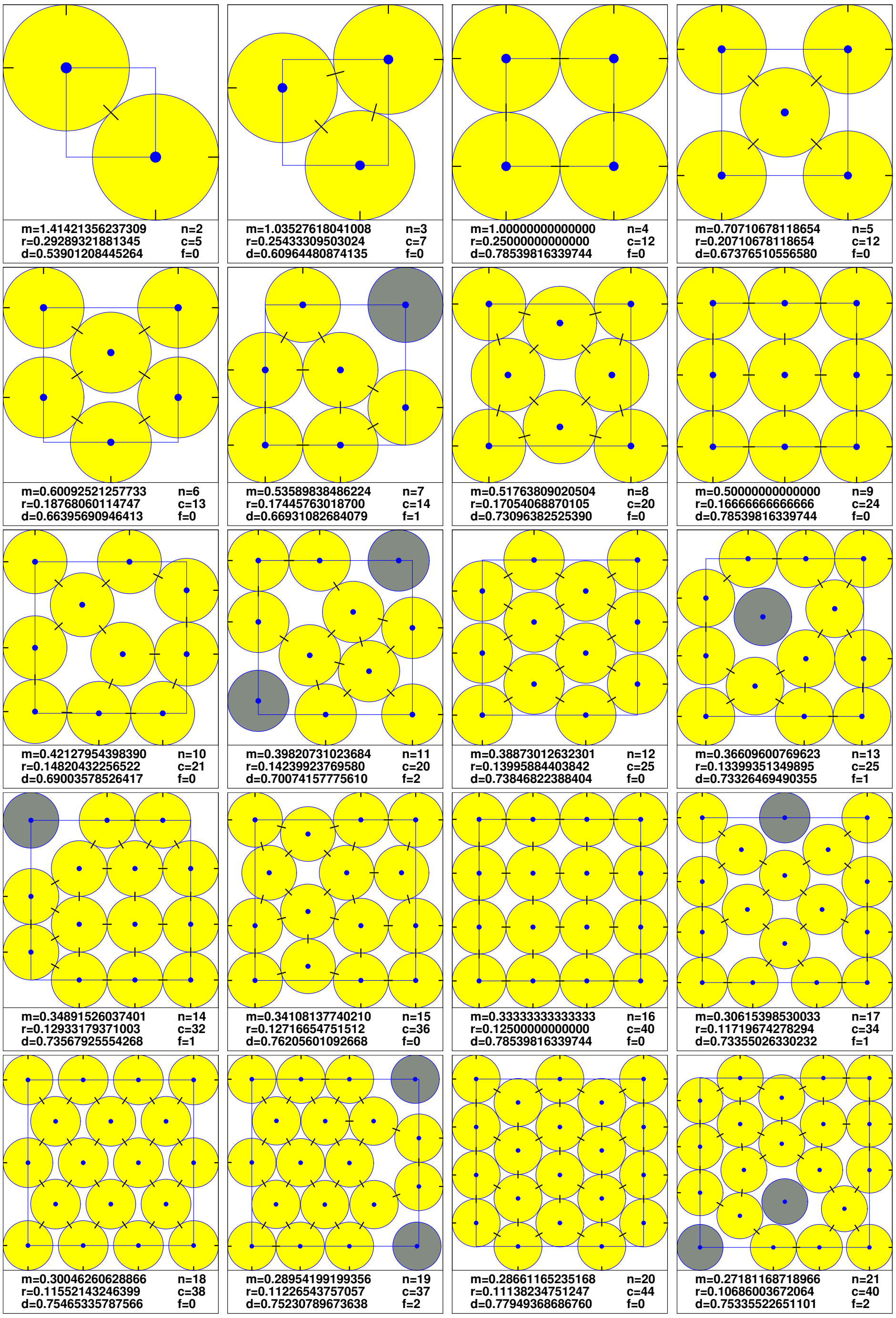

Figure 1. The best found packings for 2 – 21 circles (n is the number of circles, c the number of contacts, f the number of free circles, m the maxmin distance, r the radius of circles, and d is the density of the packing). The free circles are denoted by dark shading.

Figure 2. The best found packings for 22 – 41 circles (n is the number of circles, c the number of contacts, f the number of free circles, m the maxmin distance, r the radius of circles, and d is the density of the packing). The free circles are denoted by dark shading.

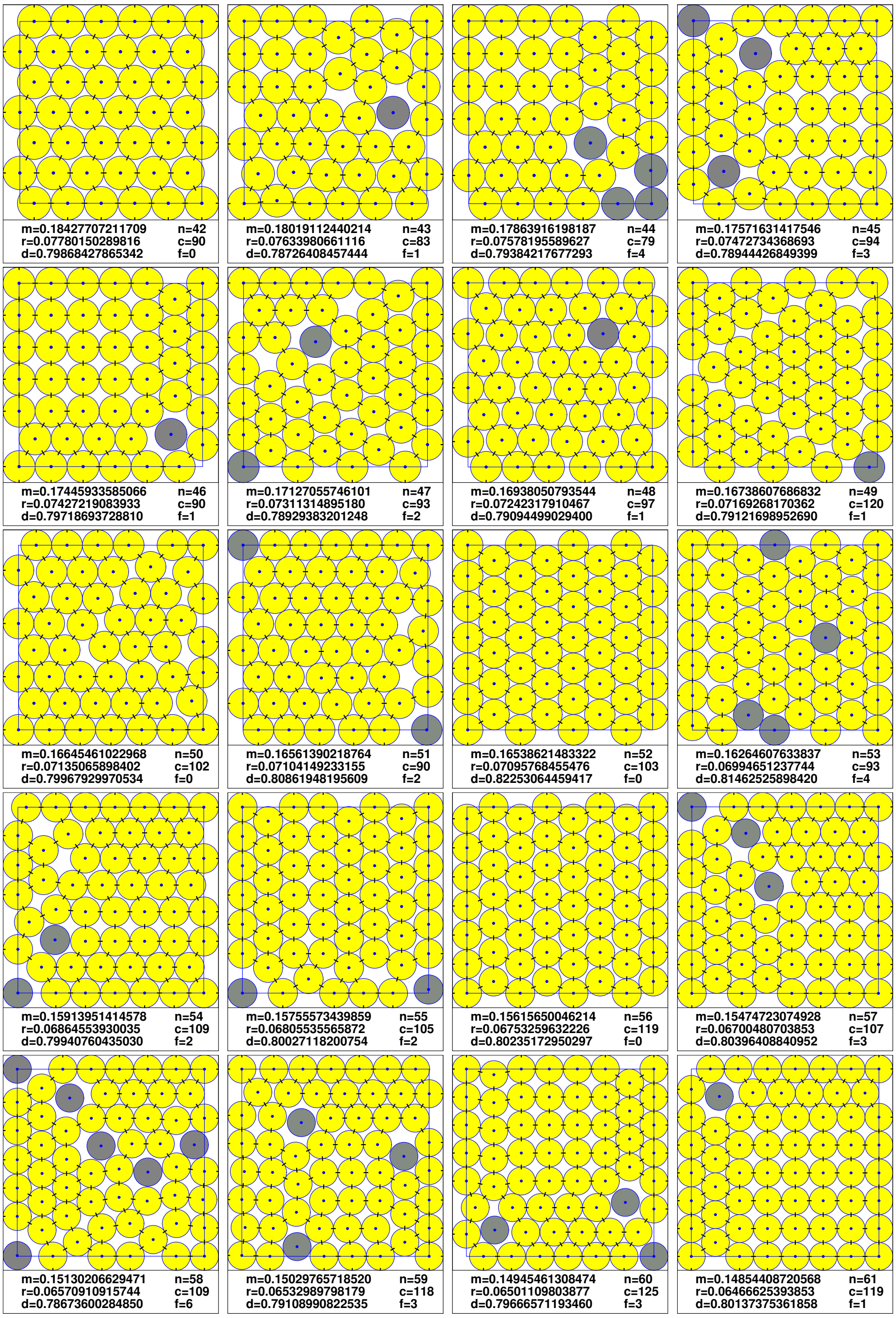

Figure 3. The best found packings for 42 – 61 circles (n is the number of circles, c the number of contacts, f the number of free circles, m the maxmin distance, r the radius of circles, and d is the density of the packing). The free circles are denoted by dark shading.

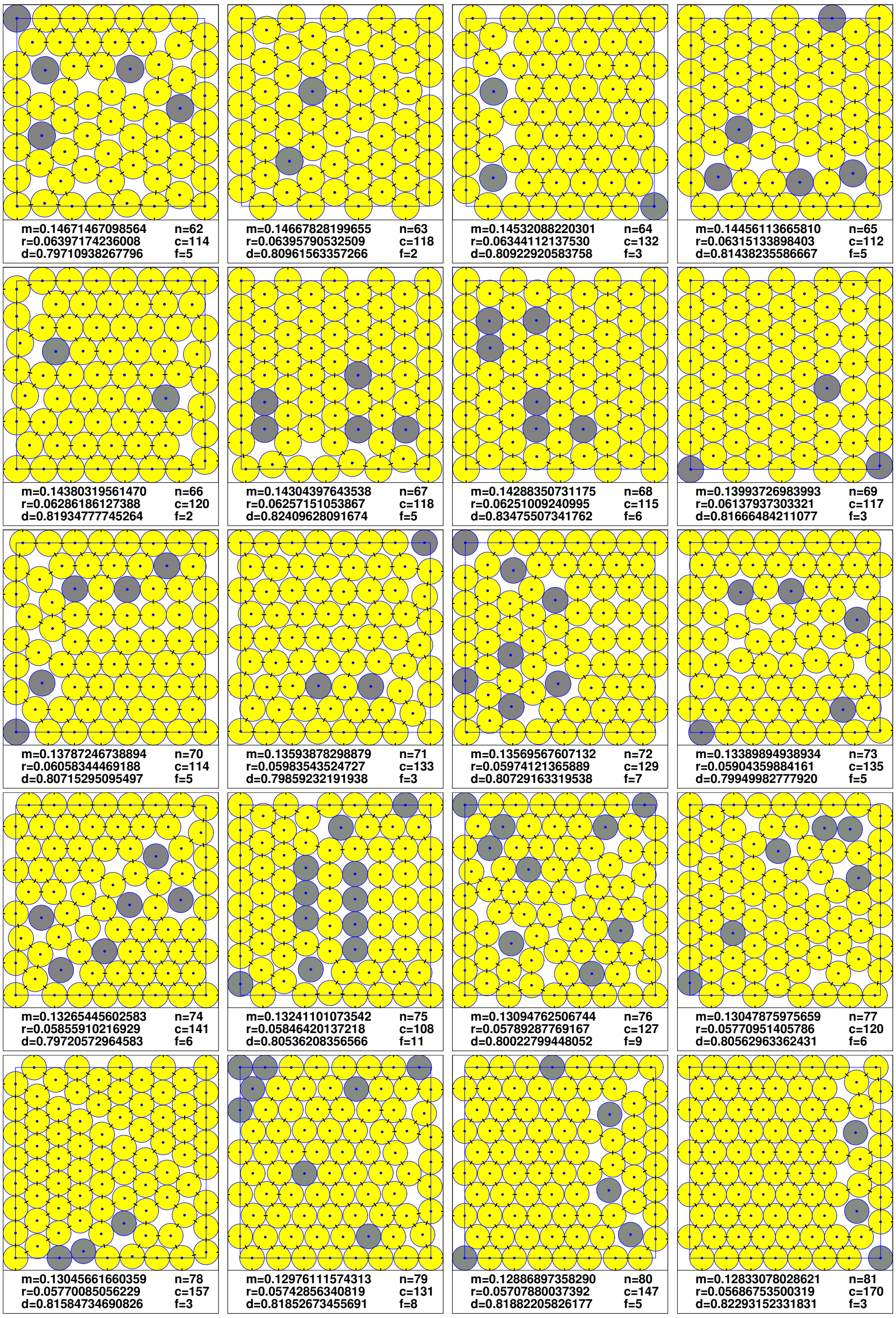

Figure 4. The best found packings for 62 – 81 circles (n is the number of circles, c the number of contacts, f the number of free circles, m the maxmin distance, r the radius of circles, and d is the density of the packing). The free circles are denoted by dark shading.

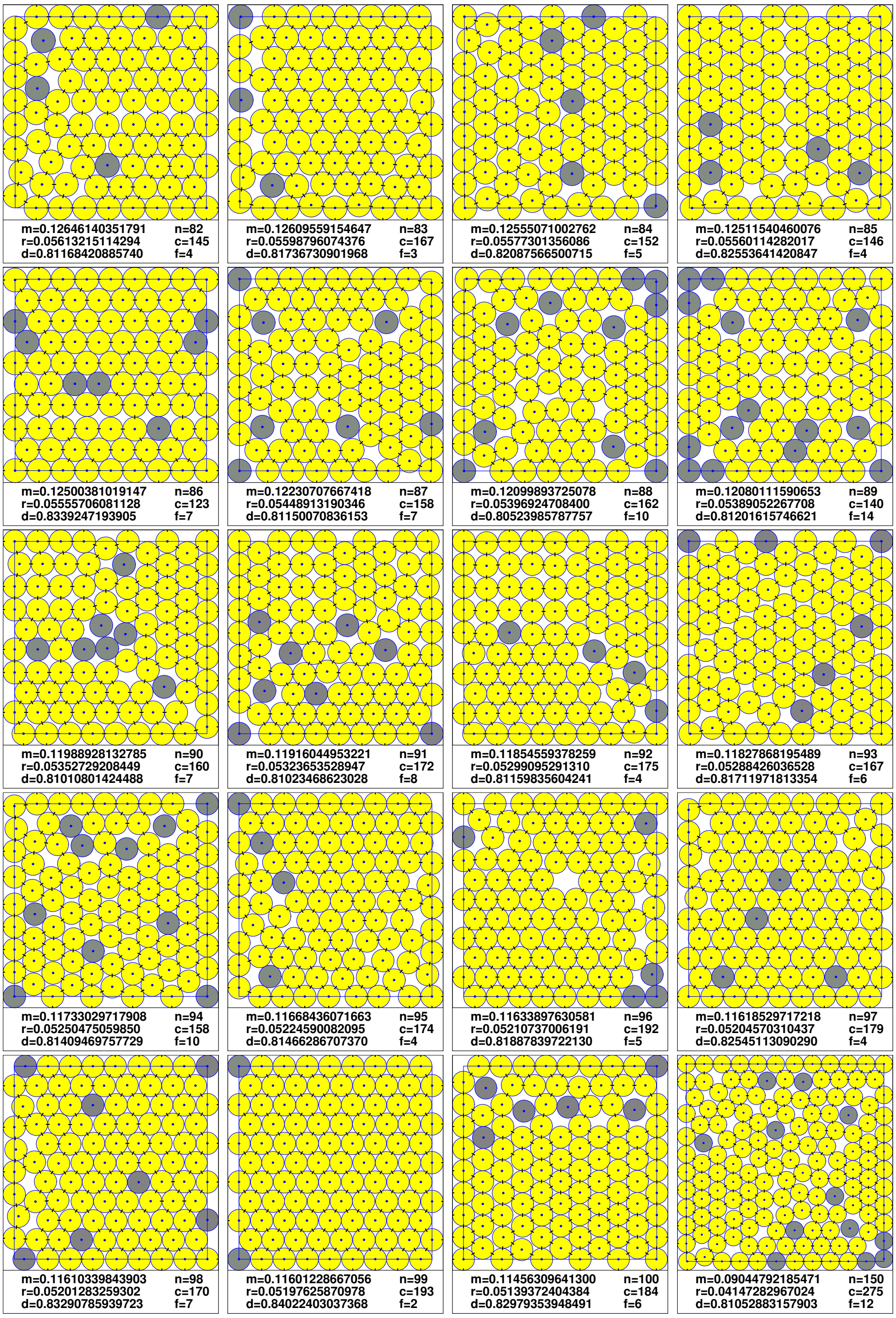

Figure 5. The best found packings for 82 – 100, and 150 circles (n is the number of circles, c the number of contacts, f the number of free circles, m the maxmin distance, r the radius of circles, and d is the density of the packing). The free circles are denoted by dark shading.

References

1. I. Althöfer and K.U. Koschnick. On the convergence of threshold accepting. *Appl. Math. Optim.*, 24:183–195, 1991.
2. G. Dueck and T. Scheuer. Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *J. Comput. Phys.*, 90:161–175, 1990.
3. R.L. Graham and B.D. Lubachevsky. Repeated patterns of dense packings of equal disks in a square. *The Electronic Journal of Combinatorics*, 3:1–16, 1996.
4. D.C. Karnop. Random search techniques for optimization problems. *Automatica*, 1:111–121, 1963.
5. M. Locatelli and U. Raber. A Deterministic Global Optimization Approach for Solving the Problem of Packing Equal Circles in a Square. *International Workshop on Global Optimization (GO.99)*, Firenze, Italy, 1999
6. C.D. Maranas, C.A. Floudas, and P.M. Pardalos. New results in the packing of equal circles in a square. *Discrete Mathematics*, 142:287–293, 1995.
7. J. Matyas. Random optimization. *Automatization and Remote Control*, 26:244–251, 1965.
8. J.R. McDonnell and D. Waagen. Evolving recurrent perceptrons for time-series modeling. *IEEE Trans. on Neural Networks*, 5(1):24–38, 1994.
9. M. Mollard and C. Payan. Some progress in the packing of equal circles in a square. *Discrete Mathematics*, 84:303–307, 1990.
10. K.J. Nurmela. Constructing combinatorial designs by local search. Series A: Research Reports 27, Digital Systems Laboratory, Helsinki University of Technology, 1993.
11. K.J. Nurmela and P.R.J. Östergård. Packing up to 50 equal circles in a square. *Discrete & Computational Geometry*, 18:111–120, 1997.
12. R. Peikert. Dichteste Packungen von gleichen Kreisen in einem Quadrat. *Elemente der Mathematik*, 49:16–26, 1994.
13. R. Peikert, D. Würtz, M. Monagan, and C. de Groot. Packing circles in a square: a review and new results. In P. Kall (ed.), *System Modelling and Optimization*, volume 180 of *Lecture Notes in Control and Information Sciences*, 45–54. Springer-Verlag, Berlin, 1992.
14. S.S. Rao. *Optimization Theory and Applications*. John Willey and Sons, New York, 1978.
15. F.J. Solis and J.B. Wets. Minimization by random search techniques. *Math. of Operations Research*, 6(1):19–50, 1981.
16. P.G. Szabó, T. Csendes, L.G. Casado and I. García,: Equal Circles Packing in a Square I. Problem Setting and Bounds for Optimal Solutions. This volume. Available at http://www.inf.u-szeged.hu/~csendes/pack1.ps.gz