Fast movement strategies for a step-and-scan wafer stepper¹

C. M. H. Kuijpers

*Department of Applied Mathematics, University of Twente, P.O. Box 217,
7500 AE Enschede, The Netherlands*

C. A. J. Hurkens*

*Eindhoven University of Technology, Department of Mathematics and
Computing Science, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*

J. B. M. Melissen

*Philips Research Laboratories, Applied Mathematics Group, Prof. Holstlaan 4,
5656 AA Eindhoven, The Netherlands*

We describe algorithms for the determination of fast movement strategies for a step-and-scan wafer stepper, a device that is used for the photolithographic processing of integrated circuits. The proposed solution strategy consists of two parts. First, we determine the maximum number of congruent rectangular chips that can be packed on a wafer, subject to the restriction that the chips are placed in a rectangular grid. Second, we find fast movement strategies for scanning all chips of a given packing, given the mechanical restrictions of the wafer stepper. The corresponding combinatorial optimization problem is formulated as a generalized asymmetric traveling salesman problem. We show how feasible scan strategies are determined, and how these strategies are improved by local search techniques, such as iterative improvement based on 2- and 3-exchanges, and simulated annealing based on 2-exchanges.

Keywords & Phrases: chip; combinatorial optimization; integrated circuit; local search; packing; photolithography; simulated annealing; traveling salesman problem.

1 Introduction

Photolithography plays an important role in the production of integrated circuits (IC's). With this technique patterns are transferred to a disc of semiconducting material (a *wafer*). The opto-mechanical system which executes this process is referred to as a *wafer stepper* (IEC, 1984).

We consider a special type of wafer stepper that operates according to the *step-and-scan* protocol (BUCKLEY and KARATZAS, 1989). An illumination system produces a slit of light, which is projected onto a semi-transparent mask, the *reticle*. The light

* wscor@win.tue.nl

¹ This paper is based on the first author's Master's thesis, which was awarded the VVS-prize 1994 of the Netherlands Society for Statistics and Operations Research.

Fig. 1. Schematic representation of the step-and-scan wafer stepper.

that filters through exposes a small portion of the photoresist-coated wafer (see Figure 1). For a correct reticle pattern transference, the reticle has to be passed under the slit of light while at the same time the wafer is moved in the opposite direction. After the complete reticle has been scanned, the wafer and the reticle are stepped to the positions required for a next exposure. The order in which parts of the wafer are scanned and the scanning directions constitute the so-called *scan strategy*.

In this paper we show how to find near-optimal scan strategies. A scan strategy is considered to be optimal if it maximizes the throughput of chips. In Section 2 we determine the maximum number of chips that can be placed on a wafer. An algorithm for computing the total processing time of a scan strategy is described in Section 3. In Section 4 we formulate the search for optimal movement strategies as a generalized asymmetric traveling salesman problem. Methods for determining a feasible scan strategy are described and algorithms are given to improve given strategies. These are iterative improvement algorithms based on 2- and 3-exchanges, and a simulated annealing algorithm based on 2-exchanges. We present some examples in Section 5 and give conclusions in Section 6.

2 Packing chips on a wafer

In designing a scan strategy that maximizes the throughput, one faces two coupled problems. First, the chips have to be placed on the wafer, and second, an optimal scan strategy for this particular arrangement of chips must be found. In theory these two problems have to be considered simultaneously, because it is possible that an arrangement with a few chips admits a very fast scan strategy, which in the end maximizes the total throughput. In practice, however, a large part of the processing time of a wafer is taken up by the loading and the alignment of the wafer. A maximum throughput can therefore only be achieved when the number of chips on the wafer is maximized. Consequently, we will treat the two problems separately.

In this section we consider the problem of finding a maximum packing. Technical requirements restrict us to packings in which the chips are arranged in a rectangular lattice. The corresponding mathematical problem is described as follows. Let C be a circular disc with radius r in the Euclidean two-space. A lattice packing in C is a collection of congruent rectangles forming part of the rectangular grid with grid constants Δx and Δy such that all rectangles are contained in C . The maximum number of rectangles of such a packing of C is denoted by $N(C, \Delta x, \Delta y)$. The problem is to calculate $N(C, \Delta x, \Delta y)$ for given r , Δx , and Δy .

No simple formulas for $N(C, \Delta x, \Delta y)$ exist. Approximations have been derived (FERRIS-PRABHU, 1989). Since we are only interested in relatively small numbers of rectangles, we will describe a polynomial-time algorithm that is guaranteed to generate a maximal lattice packing. The algorithm is based on the following more general theorem.

THEOREM 1. *Let C be a convex, compact subset of $\mathbb{R}^2$ with a non-empty interior. Then there is a maximal lattice packing of congruent rectangles in C that has at least two points in common with the boundary ∂C of C .*

PROOF. The boundedness of C implies that a maximum lattice packing P in C exists. We define P_0 to be the set of lattice points in P . First, suppose that no point of P_0 is in ∂C . There is a point in P_0 that has minimal distance to ∂C . The packing can then be

translated such that this point is on $\partial\mathcal{C}$ and all points of P_0 remain in $\mathcal{C}$. The convexity implies that all rectangles of the translated packing are also in $\mathcal{C}$.

Now suppose that exactly one point $p_0 \in P_0$ lies in $\partial\mathcal{C}$. The boundary of any convex, compact subset of $\mathbb{R}^2$ is rectifiable (BUSEMANN, 1958), so we can find a parameterization $k : [0, 1] \rightarrow \partial\mathcal{C}$ of $\partial\mathcal{C}$ such that $k(0) = k(1) = p_0$.

Let T_b denote a translation over b . For any $p \in P_0, p \neq p_0$ there is a minimal $t_p, 0 \leq t_p \leq 1$, such that $T_{k(t_p)-p_0}p \in \partial\mathcal{C}$. This can be seen as follows. Let the Euclidean inner product be denoted by $\langle \cdot, \cdot \rangle$. As $\partial\mathcal{C}$ is compact and $x \mapsto \langle x, p_0 - p \rangle$ is continuous, there is a $t_1 \in [0, 1]$ such that

$$\langle k(t_1), p_0 - p \rangle \leq \langle x, p_0 - p \rangle \text{ for all } x \in \partial\mathcal{C}$$

Since $\mathcal{C}$ is convex, the above inequality is also true for all $x \in \mathcal{C}$. Now as $p \neq p_0$,

$$\langle T_{k(t_1)-p_0}p, p_0 - p \rangle = \langle k(t_1), p_0 - p \rangle - \|p_0 - p\|^2 < \langle k(t_1), p_0 - p \rangle$$

This implies that $T_{k(t_1)-p_0}p$ is not in $\mathcal{C}$. From continuity it follows that there is a minimal $t_p \geq 0$ such that $T_{k(t_p)-p_0}p \in \partial\mathcal{C}$.

Let $t^* = \min_{p \in P_0 \setminus p_0} t_p$. Then $T_{k(t^*)-p_0}P_0$ has at least two points in common with $\partial\mathcal{C}$. Therefore, $T_{k(t^*)-p_0}P$ is a lattice packing in $\mathcal{C}$ that shares at least two points with $\partial\mathcal{C}$. $\square$

By Theorem 1 we know that for the determination of a maximum lattice packing of $\mathcal{C}$ we need only consider packings that have at least two points in common with $\partial\mathcal{C}$. The idea behind the algorithm is to generate all these packings.

Let P be a lattice packing of $\mathcal{C}$ that shares at least two points with $\partial\mathcal{C}$, and let the origin O and (p_1, p_2) be these two points. Let M be the center of $\mathcal{C}$. Since both O and (p_1, p_2) are vertices of a rectangle from the packing, we may write

$$(p_1, p_2) = (k\Delta x, l\Delta y) \text{ for some } k, l \in \mathbb{Z}$$

Due to the symmetry with respect to the horizontal and the vertical axes through M we may assume that $k, l \in \mathbb{N} \cup \{0\}$.

The two possible positions of $M = (m_1, m_2)$ are given by:

$$m_1 = \frac{1}{2} (k\Delta x - l \cdot l\Delta y) \text{ and } m_2 = \frac{1}{2} (l\Delta y + \lambda \cdot k\Delta x)$$

with

$$\lambda = \pm \sqrt{\frac{(2r)^2}{(k\Delta x)^2 + (l\Delta y)^2} - 1}$$

Using symmetry once more (rotation over 180° , interchanging the roles of O and (p_1, p_2)) we may assume that $\lambda > 0$. Thus, any choice of $r > 0$, and of $k, l \in \mathbb{N} \cup \{0\}$,

$0 < (k\Delta x)^2 + (l\Delta y)^2 \leq 4r^2$, fixes the coordinates of the center of a circular disc $C(k, l)$ of radius r .

For any circular disc C fixed in the Euclidean plane, let $N'(C)$ denote the number of rectangles within C . This number can be simply computed by calculating for each x -value $\xi = k\Delta x$, the number of y -values $\eta = l\Delta y$, for which the point (ξ, η) lies within C . From these numbers the number of rectangles within C can easily be determined. The computation can be carried out in a number of steps, bounded by $\min\{2r/\Delta x, 2r/\Delta y\}$.

We determine $N(C, \Delta x, \Delta y)$ as follows:

$$N(C, \Delta x, \Delta y) = \max_{k, l \in \mathbb{N} \cup \{0\}, 0 < (k\Delta x)^2 + (l\Delta y)^2 \leq 4r^2} N'(C(k, l)).$$

We have that

$$N(C, \Delta x, \Delta y) = \Theta\left(\frac{r^2}{\Delta x \Delta y}\right) (\Delta x^2 + \Delta y^2 \rightarrow 0)$$

as the number of rectangles inside the disc is linearly proportional to the disc area. Notice that the number of k, l combinations over which the maximum is taken is of the same order. As the computational complexity of computing N' is only the square root of this number, the total amount of work required by the algorithm is $O(N(C, \Delta x, \Delta y)^{1.5})$.

So far, we have interpreted the wafer as a perfect disc, but in reality it is truncated. However, a slightly modified application of the proof of Theorem 1 shows that again we may restrict ourselves to lattice packings with two vertices on the circular edge. Figure 2 shows an example of a maximum packing found for a truncated disc. Another point that we have left aside is the fact that for calibration and exact positioning of the wafer two alignment marks are etched on the disc. The chips in which the alignment marks are put cannot be used for IC production.

Fig. 2. A maximum lattice packing on a wafer of radius $r = 100$ mm with rectangles of size $26 \text{ mm} \times 13 \text{ mm}$.

3 Processing time of a strategy

3.1 Introduction

With the algorithm described in Section 2 we have found a maximum packing $P = \{c_i \mid i = 1, 2, \dots, N\}$ of N usable chips. Next, our purpose is to determine a time-optimal scan strategy for P . The computation of the total time required to carry out a given scan strategy involves the solution of a non-trivial optimal control problem, which is considered in this section.

A scan strategy is represented by a triple (A, π, D) , where A represents the alignment mark from which the scanning process starts, $\pi \in S_n$ is a permutation indicating the order in which the chips are scanned ($\pi_i = j$ if c_j is the i th chip to be scanned), and $D : P \rightarrow \{-1, +1\}$ denotes the scan direction of the chips ($D(c_i) = +1$ if and only if chip c_i is scanned in the positive x -direction). The j th scan is denoted by σ_j and involves the scanning of c_{π_j} .

The time required for the execution of a scan strategy for a given packing P consists of two components. First, some time is needed for the actual scanning of the chips. This time does not depend on the scan strategy, since all chips are scanned exactly once, and at a constant velocity. Second, some time is taken up by the movements of the wafer and the reticle between the scans of the individual chips. Both the wafer and the reticle need to be at the right positions and have the correct velocities at the start of a scan. The latter time strongly depends on the scan strategy.

Between two subsequent scans there are three independent movements to consider: the movements of the wafer in the x - and y -directions, respectively, and the movement of the reticle in the x -direction. Determination of the minimum time needed for each of these three movements constitutes a one-dimensional optimal control problem, the solution of which is given in the next subsection.

3.2 One-dimensional problems

Consider the movements of the wafer and the reticle before a scan. Initially, the velocity of the reticle has to be uniform and equal to its scan velocity $v_{\text{scan},r}$. At the same time the velocity of the wafer needs to be uniform and zero in the y -direction, and uniform and equal to $v_{\text{scan},wx} = (-v_{\text{scan},r}/\mu)$ in the x -direction, μ being the magnification factor of the lens.

To determine the minimum time required for the movements of the wafer in the x - and y -directions and for the movement of the reticle, we need to solve the following one-dimensional problem from mechanics: Consider the one-dimensional motion of an object, where $\varphi(t)$ denotes the position of the object at time t . Let $v(t)$ denote the velocity of the object, defined by $v(t) := \dot{\varphi}(t)$, where the dot represents the first-order derivative. Furthermore, let $a(t) := \dot{v}(t)$ denote the acceleration, and $j(t) := \dot{a}(t)$ the jerk of the object. Suppose that the mechanical system restricts the ranges of v , a , and j by $|v(t)| \leq V$, $|a(t)| \leq A$, and $|j(t)| \leq J$, for all t , where V , A and J are given parameters. What is the minimum time necessary for the object to cover a distance d if its start velocity v_{begin} and end velocity v_{end} are given and uniform, with $|v_{\text{begin}}| \leq V$ and $|v_{\text{end}}| \leq V$? Stating the problem more precisely: what is the minimum value for

T , for which there exists a function $\varphi : [0, T] \rightarrow \mathbb{R}$ with $\varphi(0) = 0$ and $\varphi(T) = d$, satisfying the above restrictions?

There are two possible scenarios in which the minimum time is achieved: either the maximum velocity $v_{\max}$ reached during the motion is equal to velocity bound V , or $v_{\max}$ is smaller than V . Note that at the moment that $v_{\max}$ is reached the velocity must be uniform.

If $v_{\max} = V$, then the motion can be described as follows. First the velocity is increased as fast as possible from v_{begin} to V . Then the velocity is kept equal to V as long as possible. Finally, as late as possible the velocity is decreased as fast as possible from V to v_{end} .

In case $v_{\max} < V$, we can describe the motion as follows: first the velocity is increased as fast as possible from v_{begin} to $v_{\max}$ ($v_{\text{begin}} \leq v_{\max} < V$), after which the velocity is reduced as quickly as possible from $v_{\max}$ to v_{end} .

We first determine the minimum time necessary to accelerate the object's movement from a uniform velocity v_1 to a uniform velocity v_2 . Without loss of generality we assume that $-V \leq v_1 \leq v_2 \leq V$. The motion that attains this minimum time can be described as follows: first the maximum jerk is exerted during a time T_1 , then the jerk is kept equal to zero for a time T_2 , and finally it is put at $-J$ for a time T_3 . Note that if during the motion the acceleration bound A is not reached, then T_2 is equal to zero. We denote the total required time by $T = T_1 + T_2 + T_3$. Let x denote the distance covered by the object in this time. For fixed bounds V , A and J , the time $T(v_1, v_2)$ and distance $x(v_1, v_2)$ depend on v_1 and v_2 only. Let φ_{v_1, v_2} denote the function for which the minimum time is achieved.

As both start and end velocity are uniform, the net increase of the acceleration is zero and hence

$$\int_0^T \frac{da}{dt} dt = \int_0^{T_1} J dt - \int_{T_1+T_2}^T J dt = J(T_1 - T_3) = 0$$

This implies that $T_1 = T_3$ and $T = 2T_1 + T_2$, giving $v_2 - v_1 = JT_1^2 + JT_1T_2$.

As $v(t) + v(T - t) = v_1 + v_2$ for all t ($0 \leq t \leq T$), the position at time T is equal to

$$\varphi(T) = \int_0^T v(t) dt = \frac{1}{2}(v_1 + v_2)T$$

If the maximum acceleration A is reached during the motion, then $v_2 - v_1 > A^2/J$, and

$$T_1 = \frac{A}{J} \text{ and } T_2 = \frac{v_2 - v_1}{JT_1} - T_1$$

Hence

$$T(v_1, v_2) = \frac{v_2 - v_1}{A} + \frac{A}{J}$$

and the distance x covered in the meantime is

$$x(v_1, v_2) = \varphi_{v_1, v_2}(T) = \frac{1}{2}(v_2 + v_1) \left(\frac{v_2 - v_1}{A} + \frac{A}{J} \right)$$

In case A is not reached, we have $v_2 - v_1 \leq A^2/J$, and

$$T_1 = \sqrt{\frac{v_2 - v_1}{J}} \text{ and } T_2 = 0$$

Then,

$$T(v_1, v_2) = 2\sqrt{\frac{v_2 - v_1}{J}}$$

and

$$x(v_1, v_2) = \varphi_{v_1, v_2}(T) = (v_1 + v_2) \sqrt{\frac{v_2 - v_1}{J}}$$

Note that for $v_1 > v_2$ we have $T(v_1, v_2) = T(v_2, v_1)$ and $x(v_1, v_2) = x(v_2, v_1)$.

We conclude that the minimum time needed to change a uniform motion with velocity v_1 into a uniform motion with velocity v_2 equals

$$T(v_1, v_2) = \begin{cases} |v_2 - v_1|/A + A/J & \text{if } |v_2 - v_1| > A^2/J \\ 2\sqrt{|v_2 - v_1|/J} & \text{otherwise} \end{cases}$$

The distance covered in the meantime is

$$x(v_1, v_2) = \begin{cases} \frac{1}{2}(v_2 + v_1)(|v_2 - v_1|/A + A/J) & \text{if } |v_2 - v_1| > A^2/J \\ (v_2 + v_1)\sqrt{|v_2 - v_1|/J} & \text{otherwise} \end{cases}$$

With the above formulas we can compute the distance covered by an object that starts at a uniform velocity of value v_{begin} , then increases its velocity as fast as possible to a uniform value v_{max} , and after that decreases its velocity as fast as possible to a uniform value v_{end} . For given v_{begin} , v_{end} and v_{max} , the distance covered is $x(v_{\text{begin}}, v_{\text{max}}) + x(v_{\text{max}}, v_{\text{end}})$.

We use this to solve the inverse problem: given a start velocity v_{begin} , an end velocity v_{end} , and a distance D to cover, what is the minimum time in which this movement can be performed, and what is the maximum velocity attained during this motion?

Without loss of generality, we assume that D is positive. If $D \geq x(v_{\text{begin}}, V) + x(V, v_{\text{end}})$, then obviously the maximum velocity attained is V . The minimum time necessary is then given by

$$T(v_{\text{begin}}, V) + T(V, v_{\text{end}}) + \frac{D - x(v_{\text{begin}}, V) - x(V, v_{\text{end}})}{V}$$

Otherwise, the maximum velocity v_{max} is found by a bisection algorithm that searches for a value of v_{max} in the interval $(\max\{v_{\text{begin}}, v_{\text{end}}\}, V)$ satisfying $x(v_{\text{begin}}, v_{\text{max}}) + x(v_{\text{max}}, v_{\text{end}}) = D$. In this case the minimum time is given by $T(v_{\text{begin}}, v_{\text{max}}) +$

$T(v_{\max}, v_{\text{end}})$. Note that such a value of $v_{\max}$ exists if $x(v_{\text{begin}}, v_{\text{end}}) \leq D < x(v_{\text{begin}}, V) + x(V, v_{\text{end}})$. If $x(v_{\text{begin}}, v_{\text{end}}) > D$, then obviously the motion has to be slowed down. The bisection algorithm is then used to find the minimum velocity attained during the motion.

3.3 Settling times

Let us consider again the movements of the wafer and the reticle before a scan. With the results from the previous section we can determine the minimum time needed for the reticle to move to the right position and proceed with a scan velocity $v_{\text{scan},r}$ again. However, at the time the desired velocity is reached, the system is unsteady. Some time is required to assure that disturbances and aberrations in the velocity have faded out. This time is called the settling time t_r for the reticle. For the same reason, the wafer needs a settling time t_{wx} for the movement in the x -direction and t_{wy} for the y -direction. These settling times are considered to be given parameters, independent of the prior motions. During its settling time the reticle covers a distance $s_r = |t_r v_{\text{scan},r}|$. Hence, the uniform velocity $v_{\text{scan},r}$ must already be reached at a distance s_r ahead of the position from which the scan starts. Analogously, the wafer has to reach $v_{\text{scan},wx}$ at a distance $s_x = |t_{wx} v_{\text{scan},wx}|$ in the x -direction ahead of the position from which the scan starts.

One implication of taking settling times into account is that the distance covered in the motion from chip c_i to chip c_j ($i \neq j$) with scan directions $D(c_i)$ and $D(c_j)$ is in general different from the distance covered in the motion from chip c_j to chip c_i with reversed scan directions. Figure 3 indicates how the presence of settling times leads to asymmetric time distances: after accounting for the distance covered during the settling period, the remaining net distances are not equal anymore.

Fig. 3. The settling times lead to asymmetric distances.

3.4 Three-dimensional problem

We now consider the problem of determining the actual minimum time between subsequent scans σ and τ . Let $\mathcal{T}_r$ be the minimum time in which the reticle can perform its movement between scans σ and τ . Analogously, we define $\mathcal{T}_{wx}$ and $\mathcal{T}_{wy}$ to be the minimum time required for the wafer to move to the right positions in the x -direction, and the y -direction, respectively. One would expect that the minimum time required between the scans σ and τ would be equal to $\max\{\mathcal{T}_r, \mathcal{T}_{wx}, \mathcal{T}_{wy}\}$. Closer inspection reveals that this is not always true, since the required movement of the wafer cannot always be slowed down arbitrarily. This can be seen as follows.

Consider, for example, the movement of the wafer between two subsequent scans for the special case that both scans are in the positive x -direction and that the distance δ to be covered by the wafer in the positive x -direction is very small. Suppose that $0 < \delta < x(v_{\text{scan},wx}, 0) + x(0, v_{\text{scan},wx})$. The fastest way to perform this motion is to accelerate as fast as possible until the first half of the distance has been covered and apply maximum deceleration during the rest of the motion. Let α denote the time required.

If we allow the wafer to move in the positive x -direction only, the slowest way in which the motion can be performed is obtained by first decelerating until $\delta/2$ is covered and next accelerating as fast as possible during the rest of the motion. Let β denote the time for the motion in this scenario. The actual motion can be realized in any time between α and β .

If we allow the wafer to move in the negative x -direction as well, then distance δ can be covered even more slowly as follows. First, the wafer moves some time in the positive x -direction, while decelerating. Next, it moves in the negative x -direction, accelerating until a net distance of $\delta/2$ has been covered. Then, the movement in the negative direction is slowed down, and finally the wafer moves in the positive x -direction again. Let γ denote the minimum time in which such a movement is carried out. The bisection algorithm described in the previous section is used to compute γ . The movement described here can be slowed down so that it takes a time in the interval $[\gamma, \infty)$.

The set of times in which distance δ can be covered is the union of intervals $[\alpha, \beta] \cup [\gamma, \infty)$. Special care has to be taken in case $\beta < \gamma$.

4 Improvement of strategies

The problem of determining an optimal scan strategy for a given packing $P = \{c_i \mid i = 1, 2, \dots, N\}$ can be converted into a generalized asymmetric traveling salesman problem in the following way.

We represent every chip c_i by two points, v_{i1} and v_{i2} , as shown in Figure 4. If v_{i1} is visited before v_{i2} , then chip c_i is scanned in the positive x -direction; otherwise it is scanned in the negative x -direction. Let v_{01} and v_{02} represent the alignment marks, and v_{N+1} the end point of the scan. For reasons of accuracy, the scanning has to start from either one of the alignment marks. Furthermore, after the scanning of the chips has been completed, the wafer has to move to a neutral position. This corresponds to a final move from the last chip scanned, to a fixed position v_{N+1} , outside of the disc area.

We define a set of vertices V , a set of edges E and a set of arcs A as follows:

$$\begin{aligned} V &:= \{v_{ij} \mid i = 0, 1, \dots, N, j = 1, 2\} \cup \{v_{N+1}\} \\ E &:= \{\{v_{i1}, v_{i2}\} \mid i = 1, 2, \dots, N\} \\ A &:= \{(v_{0j}, v_{kl}) \mid j, l = 1, 2, k = 1, 2, \dots, N\} \\ &\quad \cup \{(v_{ij}, v_{kl}) \mid j, l = 1, 2, i, k = 1, 2, \dots, N, i \neq k\} \\ &\quad \cup \{(v_{ij}, v_{N+1}) \mid j = 1, 2, i = 1, 2, \dots, N\} \end{aligned}$$

Fig. 4. There are four arcs from chip c_i to chip c_j .

Fig. 5. Schematic representation of a feasible path.

The scan problem is now defined as follows: find a path of minimum cost in the graph $G = (V, E, A)$, which starts in v_{01} or in v_{02} , further visits all vertices of $V \setminus \{v_{01}, v_{02}, v_{N+1}\}$ exactly once, ends in v_{N+1} , and traverses each edge of E . E is referred to as the set of the *obligatory edges*. The problem described above is a mixed rural postman problem (ORLOFF, 1974; BENAVENT et al., 1985).

The problem is tackled in the following way. First, a feasible scan strategy (see Figure 5) is created. Many methods exist for doing this (LAWLER et al., 1985). In Figure 6(a) an algorithm is described for generating a feasible strategy at random. Better feasible paths may be created with the *insertion* algorithm given in Figure 6(b), since it uses information about the cost.

Random Algorithm	Insertion Algorithm
<p>Step 0. Choose at random an alignment mark from which to start.</p> <p>Step 1. Generate a random permutation π of $\{1, 2, \dots, N\}$. If $\pi_i = j$, then c_j is the ith chip to be scanned.</p> <p>Step 2. For every chip, choose a scan direction at random.</p>	<p>Step 0. Start with a partial path consisting of one arbitrarily chosen alignment mark v_{0x}, and v_{N+1}. Set $k = 1$.</p> <p>Step 1. Insert v_{k1} and v_{k2} as cheaply as possible, so that the resulting path represents a feasible scan strategy of the chips $c_1, c_2, \dots, c_k$.</p> <p>Step 2. If $k < N$, set $k := k + 1$ and go to step 1; otherwise stop.</p>
(a)	(b)

Fig. 6. Two algorithms for generating a feasible path.

Fig. 7. All of the 2-exchanges involving the first arc of the path (cf. Figure 5).

The generated feasible paths are subject to local search algorithms (LAWLER et al., 1985; PAPADIMITRIOU and STEIGLITZ, 1982) such as iterative improvement based on 2-exchanges (CROES, 1958) and 3-exchanges (LIN, 1965), and a simulated annealing algorithm (KIRKPATRICK et al., 1983; AARTS et al., 1988, 1989) based on 2-exchanges. The algorithms for improving a scan strategy differ from the standard algorithms in the following. First, the obligatory edges E are not eligible for an exchange. Second, in this context a 2-exchange refers to selection of two arcs that leave the current path. By this, the path is split into three parts. These are recombined to a feasible path by visiting the middle part in reverse order. As the distances are not symmetric, such a “2-exchange” actually involves the exchange of more than two arcs. Third, in case the first arc of the feasible path is selected for an exchange, it is also possible to just change the starting point of the path (see Figure 7).

5 Results, examples

In this section we consider two characteristic examples. They are characteristic in the sense that the “optimal” patterns that we found are quite different. On the other hand there is a particular similarity. The examples show how the outcome is influenced by the settling times. It turns out that the average travel time between two subsequent scan in the final solution is approximately 0.122, which is about ten times the average settling time. We first describe the results. An explanation is then given in the final section.

5.1 Example 1

First, we consider the maximum packing shown in Figure 8 in combination with the settling times $t_{wx} = t_{wy} = t_r = 0.01$. All time distances are given in seconds. The

Fig. 8. An optimal packing of a wafer with radius $r = 70$ mm with 36 chips of size $20\text{ mm} \times 15\text{ mm}$.

shaded regions in Figure 8 denote chips that are unusable, as they contain the alignment marks.

With the iterative improvement algorithms based on 2- and 3-exchanges, we found the results shown in Table 1. Table 2 gives results obtained by the simulated annealing algorithm. The length of a strategy is given in seconds. All experiments were done on a SparcServer 1000 under operating system Solaris 2.3.

One run of the iterative improvement algorithm based on 2-exchanges took about 0.18 second, while an execution of the algorithm based on 3-exchanges needed about 1.88 seconds. One execution of the simulated annealing algorithm took between 1 and 6 minutes. Several strategies of length 4.270360 were found. One of them is shown in Figure 9. We see that all three algorithms found this minimum value. Of course, this is merely an indication that the end result is close to optimal. It should not be interpreted as an actual proof.

Table 1. Example 1: results obtained with iterative improvement.

algorithm	number of trials	best strategy	average strategy	worst strategy
iterative improvement based on 2-exchanges	50	4.344681	4.445710	4.660919
	100	4.319270	4.440255	4.660919
	250	4.288648	4.433182	4.660919
	500	4.288648	4.435662	4.660919
	1000	4.288648	4.435027	4.803392
	5000	4.270360	4.431938	4.803392
	10000	4.270360	4.430636	4.803392
iterative improvement based on 3-exchanges	50	4.283491	4.373497	4.449680
	100	4.283491	4.372027	4.554897
	250	4.283491	4.371650	4.554897
	500	4.280684	4.368450	4.554897
	1000	4.279078	4.367560	4.554897
	5000	4.270360	4.366698	4.566241

Fig. 9. Example 1: one of the best scan strategies found.

Table 2. Example 1: some results obtained with simulated annealing.

parameters $t = 0.005$, $NT_{\text{steps}} = 100$, 25 trials		best strategy	average strategy	worst strategy	average running time
TFact = 0.90	nover = 5n, nlimit = n	4.284289	4.333674	4.402729	1.00 min.
	nover = 5n, nlimit = 2n	4.280684	4.332035	4.458880	1.26 min.
	nover = 5n, nlimit = 3n	4.283437	4.319891	4.393454	1.44 min.
	nover = 10n, nlimit = n	4.287796	4.349301	4.424076	1.62 min.
	nover = 10n, nlimit = 2n	4.279078	4.326003	4.458880	2.02 min.
	nover = 10n, nlimit = 3n	4.283437	4.322097	4.393454	4.27 min.
	nover = 15n, nlimit = n	4.305341	4.347662	4.402729	2.33 min.
	nover = 15n, nlimit = 2n	4.283437	4.323857	4.458880	3.10 min.
TFact = 0.95	nover = 15n, nlimit = 3n	4.279078	4.326887	4.380268	3.04 min.
	nover = 5n, nlimit = n	4.279078	4.327398	4.393454	1.81 min.
	nover = 5n, nlimit = 2n	4.283437	4.315967	4.375909	2.48 min.
	nover = 5n, nlimit = 3n	4.284289	4.324337	4.381902	2.66 min.
	nover = 10n, nlimit = n	4.279078	4.331399	4.390498	2.98 min.
	nover = 10n, nlimit = 2n	4.279078	4.324323	4.389095	4.68 min.
	nover = 10n, nlimit = 3n	4.274719	4.311136	4.368043	4.88 min.
	nover = 15n, nlimit = n	4.274719	4.322314	4.389095	4.50 min.
TFact = 0.99	nover = 15n, nlimit = 2n	4.283437	4.318696	4.371550	5.42 min.
	nover = 15n, nlimit = 3n	4.279078	4.317620	4.371550	5.91 min.
	nover = 5n, nlimit = n	4.270360	4.311213	4.362832	2.03 min.
	nover = 5n, nlimit = 2n	4.279078	4.303401	4.362832	2.51 min.
	nover = 5n, nlimit = 3n	4.270360	4.300221	4.367191	3.49 min.
	nover = 10n, nlimit = n	4.270360	4.310197	4.362832	1.28 min.
	nover = 10n, nlimit = 2n	4.279930	4.316739	4.362832	2.31 min.
	nover = 10n, nlimit = 3n	4.274719	4.309454	4.358975	3.48 min.
	nover = 15n, nlimit = n	4.270360	4.313869	4.372347	1.79 min.
	nover = 15n, nlimit = 2n	4.279930	4.317615	4.362832	2.30 min.
	nover = 15n, nlimit = 3n	4.274719	4.309454	4.358975	3.48 min.

t : the initial temperature.

NT_{steps} : the number of temperature steps tried.

TFact: the factor by which the temperature is reduced on every step.

nover: the maximum number of exchanges tried at any temperature.

nlimit: the maximum number of successful exchanges during one temperature step.

Fig. 10. Example 2: best scan strategy found (with iterative improvement).

5.2 Example 2

We consider the same optimal packing as in Example 1, but now in combination with the following settling times: $t_{wx} = 0.05$, $t_{wy} = 0.01$, $t_r = 0.005$. With the iterative improvement algorithms based on 2- and 3-exchanges, we found the results given in Table 3. This time an execution of the algorithm based on 2-exchanges took 0.14 second, while the one based on 3-exchanges took 3.47 seconds. A zigzag strategy appears to be favorable here (see Figure 10).

6 Conclusions

We decomposed the problem of finding a time-optimal movement strategy for a step-and-scan wafer stepper into three subproblems. The first subproblem was the determination of a maximum packing of a given wafer. The second subproblem was the calculation of the cost (duration) of a given movement strategy. For this, it was

Table 3. Example 2: results obtained with iterative improvement.

algorithm	number of trials	best strategy	average strategy	worst strategy
iterative improvement based on 2-exchanges	50	4.924045	5.128563	5.345983
	100	4.885395	5.148303	5.499273
	250	4.885395	5.148635	5.499273
	500	4.867493	5.153122	5.499273
	1000	4.867493	5.149256	5.664390
	5000	4.831165	5.146263	5.664390
	10000	4.831165	5.144634	5.767070
iterative improvement based on 3-exchanges	50	4.868983	4.971365	5.112126
	100	4.865444	4.962240	5.199053
	250	4.819380	4.966832	5.199053
	500	4.819380	4.965938	5.202070
	1000	4.819380	4.967142	5.203557
	5000	4.819380	4.969288	5.215610

necessary to analyze the motions made by the wafer and the reticle between two subsequent scans. Finally, the scanning problem was modelled as a traveling salesman problem. We tackled this problem with some standard heuristics. Among these heuristics were iterative improvement algorithms based on 2- and 3-exchanges, both combined with multiple starts on random initial paths. Also a simulated annealing algorithm was developed, employing a neighborhood structure based on 2-exchanges.

The algorithms were implemented and computational experiments were carried out on some real-life instances. Main goal of these experiments was to see how the optimal solutions and patterns depend on the mechanical parameters, in particular on the settling times. Another goal was to see whether spending extra computing time with the simulated annealing approach would yield better results. Only some of the experimental results are mentioned in the paper. From the results we have drawn the following conclusions:

1. The scan strategies found with the algorithms described usually contain patterns that are repeated several times. Subsequent chips are always scanned in opposite directions. This can be explained by the amount of time necessary to reposition the reticle in case of two subsequent scans in the same direction. Let α denote the average traveling time between two subsequent scans in the final solution. Then positioning the reticle for a reverse scan would take approximately 0.5α to 0.8α , whereas repositioning for a scan in the same direction would take 3α !
2. Subsequent scans are usually carried out either on chips right next to each other, or above one another. That is because moving a single row or column takes time approximately α . Larger movements involve times at least 1.5α .
3. In the first example a movement in the x -direction was slightly preferred over a move in the y -direction and this leads to a pattern in which chips are scanned in a row-oriented fashion. In another experiment (not described here) high values for the settling time in the y -direction would lead to a pattern with row-by-row scanning.
4. In case the settling time of the wafer in the x -direction is larger than in the y -direction, we often find that two subsequent scans concern two chips in the same column, as shown in the second example.
5. A single run of the iterative improvement methods may easily lead to poor examples as is found by comparing results in Tables 1 and 3.
6. The simulated annealing approach does not really pay off. In the time of a single run, one can easily run one hundred trials of the iterative improvement algorithm. Picking the best result outperforms the average simulated annealing result.

Acknowledgements

We would like to thank ir. F.B. Sperling for providing technical information and dr. A.J.E.M. Janssen for discussions concerning the derivation of the time distances. The first author wishes to thank her supervisor prof.dr. J.K. Lenstra.

References

- AARTS, E. H. L., J. H. M. KORST and P. J. M. VAN LAARHOVEN (1988), A quantitative analysis of the simulated annealing algorithm: a case study for the traveling salesman problem, *Journal of Statistical Physics* **50**, 187–206.
- AARTS, E. H. L. and J. H. M. KORST (1989), *Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing*, Wiley, Chichester, 3–93.
- BENAVENT, E., V. CAMPOS, A. CORBERÁN and E. MOTA (1985), Análisis de heurísticos para el problema del cartero rural, *Trabajos de Estadística y de Investigación Operativa* **36**, 27–38.
- BUCKLEY, J. D. and C. KARATZAS (1989), Step and scan: A systems overview of a new lithography tool, *SPIE, Vol. 1088, Optical/Laser Microlithography II*, 424–433.
- BUSEMANN, H. (1958), *Convex surfaces*, Interscience Publishers, New York.
- CROES, A. (1958), A method for solving traveling-salesman problems, *Operations Research* **5**, 791–812.
- FERRIS-PRABHU, A. V. (1989), An algebraic expression to count the number of chips on a wafer, *IEEE Circuits and Devices Magazine* **5**, 37–39.
- IEC (1984), *Practical integrated circuit fabrication*, Integrated Circuit Engineering Corporation, Scottsdale.
- KIRKPATRICK, S., C. D. GELATT and M. P. VECCHI, (1983), Optimization by simulated annealing, *Science* **220**, 671–680.
- KUIJPERS, C. M. H. (1993), *Determination of a time-optimal movement strategy for a step-and-scan wafer stepper*, Master's thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology.
- LAWLER, E. L., J. K. LENSTRA, A. H. G. RINNOOY KAN and D. B. SHMOYS (eds.) (1985), *The traveling salesman problem: a guided tour of combinatorial optimization*, Wiley, Chichester, 145–160, 215–222, 177–178.
- LIN, S. (1965), Computer solutions of the traveling salesman problem, *Bell System Technical Journal* **44**, 2245–2269.
- ORLOFF, C. S. (1974), A fundamental problem in vehicle routing, *Networks* **4**, 35–64.
- PAPADIMITRIOU, C. H. and K. STEIGLITZ (1982), *Combinatorial optimization: algorithms and complexity*, Prentice-Hall, Englewood Cliffs, NJ, 454–486.

Received: January 1995. Revised: January 1996.