Chen *et al. Mathematics-in-Industry Case Studies* (2017) 8:4
DOI 10.1186/s40929-017-0012-0

Mathematics-in-Industry
Case Studies

RESEARCH Open Access

Modelling and optimizing a system for testing electronic circuit boards

Stephen Y. Chen[1], Odile Marcotte[2*], Mario Leonardo Morfin Ramírez[3] and Mary Pugh[4]

*Correspondence:
Odile.Marcotte@gerad.ca
[2]GERAD, HEC Montréal and Département d'informatique, UQAM, 3000 Côte-Sainte-Catherine, Montréal H3T 2A7, Canada
Full list of author information is available at the end of the article

Abstract
In this article we consider a difficult combinatorial optimization problem arising from the operation of a system for testing electronic circuit boards (ECB). This problem was proposed to us by a company that makes a system for testing ECBs and is looking for an efficient way of planning the tests on any given ECB. Because of its difficulty, we first split the problem into a covering subproblem and a sequencing subproblem. We also give a global formulation of the test planning problem. Then we present and discuss results pertaining to the covering and sequencing subproblems. These results demonstrate that their solution yields testing plans that are much better than those currently used by the company. Finally we conclude our article by outlining avenues for future research.

Keywords: Electronic circuit board, Combinatorial optimization, Covering, Mixed integer programming, Travelling salesman problem

Introduction

In this article we study the operations of a system for testing Electronic Circuit Boards (ECBs) that uses flying probes. Our study was carried out in cooperation with a company that will be referred to as *the Company* in what follows. We first describe the broad outlines of this system (called the *FP system*); further details will be given in the following sections. The FP system includes eight *shuttles*: four shuttles that are above the board and four shuttles that are below the board. To each shuttle are attached some *probes* (usually two or three probes). The board being tested contains *nets*, each of which can be viewed as a wire with a finite number of *points of interest* (or simply points). For the purposes of this study, a net is a set of points whose coordinates are known. Note that the nets are pairwise disjoint, i.e., no point belongs to more than one net.

Given a subset R of nets, a *test* consists of assigning certain probes to nets in R so that one point in each net is touched by one probe; the set of probes that may be used to touch a given point is known in advance and depends upon the test being carried out. Hence a test consists, formally, of a collection of nets and a family of probe sets (one set of probes for each point in each net involved in the test). The probes must touch the points simultaneously, which means that there must be a matching between a subset Q of the probes and the collection R of nets. Of course such a matching need not exist: its existence depends upon the locations of the eight shuttles. Figure 1 displays two tests, one of which can be carried out in the current configuration but the other cannot.

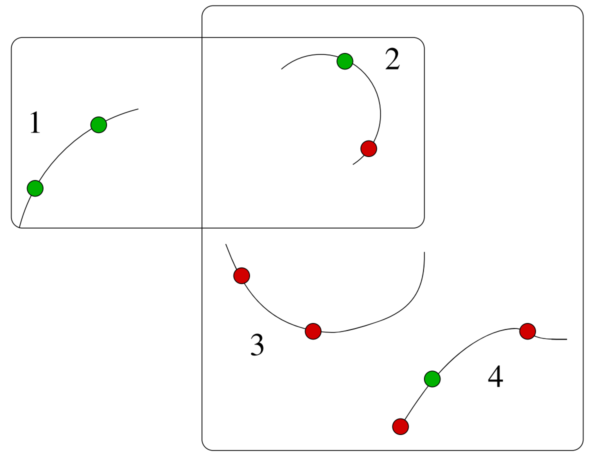

Fig. 1 Four nets involved in two tests. The points of interest pictured in green are those that can be touched by at least one probe. Those pictured in red cannot be touched by any probe. The test consisting of nets 1 and 2 can be carried out in the current configuration but not the test consisting of nets 2, 3, and 4

In general, given a *configuration of shuttles*, only a fraction of the tests can be carried out. Thus several configurations must be computed in order to ensure that all the tests are covered. (Note that the initial configuration, which is also the final one, is given in advance.) Moving the shuttles from one configuration to the next requires some time, which can be evaluated if one knows the coordinates of the shuttles within their successive configurations. The objective of the Company is to minimize the total time needed to move the shuttles between configurations, with the constraint that each test is covered by at least one configuration. The system currently used by the Company yields poor results, in the sense that the number of configurations needed to cover all the tests is far too large, resulting in many unnecessary shuttle moves. Hence the problem was proposed to us by the Company at the Fields-MITACS Industrial Problem-Solving Workshop (see Chen S, Gustafsson J, Marcotte O, Morfin M, Pugh M: Improved optimization for a testing system with two mobility layers, in Proceedings of the Fields-MITACS Industrial Problem-Solving Workshop, forthcoming).

The *planning problem* just described is similar to the Travelling Salesman Problem (TSP), one of the basic problems in combinatorial optimization (see Applegate et al. [2]). We conjecture that in its general form, the planning problem is as difficult to solve as the TSP. Although we are not aware of any previous work on the planning problem, there are similarities between our problem and two classes of problems that have been considered already: geometric covering problems (see for instance the article by Arkin and Hassin [3] and the article by Ahn et al. [1]) and some generalizations of the TSP. One of these generalizations is the Travelling Politician Problem (see Riordan [19]). Other generalizations include the Covering Salesman Problem (see the article by Current and Schilling [5]) and the Clustered Travelling Salesman Problem (CTSP), discussed in an article by Laporte and Palekar [11] that includes an application of the CTSP to a circuit design problem.

The Generalized Travelling Salesman Problem (GTSP) has received a fair amount of attention (see for example the articles [12–15] by Laporte and his colleagues). Some authors have proposed transforming a GTSP instance into a TSP instance (for an example see Dimitrijevic and Saric [6]). Others have proposed a Lagrangian-based approach for the asymmetric GTSP (Noon and Bean [17]) or a branch-and-cut algorithm for the symmetric GTSP (Fischetti et al. [8]). Fischetti et al. [7] have studied the symmetric GTSP polytope. Recently Pop [18] and Kara, Guden, and Koc [10] have proposed new formulations for the GTSP. In spite of the similarities between the planning problem and the various problems we have just mentioned, our problem seems to be completely new and a major portion of the present article will be devoted to its formulation as a mathematical program.

As mentioned before, although the objective of the Company is to minimize the total time needed to move the shuttles between configurations, the main difficulty faced by the Company is the large number of configurations required to cover all the tests in the system that it currently uses. Hence our first objective is to try to reduce the number of configurations. To achieve this we tackle the planning problem in two steps. First we construct a partition of the set of tests into subsets that can be carried out within a single configuration. We call this problem the *covering problem* since we are looking for a family of configurations that covers each and every test. Then we solve the *optimal sequence problem* in order to find the best sequence of configurations (using only those configurations that are in the solution of the covering problem). Since the decomposition of the planning problem into a covering problem and a sequencing problem does not always yield an optimal solution, we also present a global formulation of the planning problem.

Our article is organized as follows. The "Methods" section deals with the modelling and solution of the planning problem. It includes a subsection on shuttles and probes; a subsection on power chains and avoidance of collisions; a subsection on probes, points of interest, and reachability constraints; subsections on incompatibility constraints and tests and feasibility constraints, respectively; a subsection on the partial covering problem; a subsection on a greedy algorithm for the covering problem; a subsection on the optimal sequence problem; and finally a subsection on a global formulation of the planning problem. The article continues with a section on "Results and discussion" and ends with the "Conclusion" section.

Methods

Shuttles and probes

We assign the indices $0, 1, \ldots, 7$ to the eight *shuttles* in the following way: the shuttles that are above the board are labelled 0, 1, 2, 3 in the clockwise order starting with the front-left shuttle and the shuttles below are labelled 4, 5, 6, 7 in a similar way. Hence Shuttle 0 is the front-left shuttle that is above the board, Shuttle 1 is the back-left shuttle above the board, Shuttle 2 the back-right shuttle above the board, and so on. The dimensions of each shuttle are 195 mm (in the horizontal direction) and 160 mm (in the vertical direction). The dimensions of the board are 1050 mm (in the horizontal direction) and 850 mm (in the vertical direction). We will use these constants in the model but obviously they can be replaced by other constants if the covering problem must be solved for another system. Each shuttle has a *preferred corner*, corresponding to the point where its *power chain* is attached to the board. For instance, the preferred corner of the front-left shuttle that is above (resp. below) the board is the front-left corner of this shuttle.

The *initial configuration* is the configuration in which the preferred corner of each shuttle is located at the corresponding board corner. For instance, the initial coordinates of the preferred corner of the back-left shuttle are $(0, 850)$. A given board cannot be introduced into (or removed from) the FP system unless it is in the initial configuration. Hence when the board is introduced into the FP system, one computes the tests that can be carried out within this configuration before considering other configurations.

In our model the x-coordinate of Shuttle k (for $k \in \{0, 1, \ldots, 7\}$) will be denoted by S_{2k} and its y-coordinate by S_{2k+1}: those two expressions are variables since we are looking for "good" configurations and the goal of the model is to find them. The *probes*, which can come into contact with *points of interest* (or points, for short), are attached to their respective shuttles. The number of probes in the system under consideration is 21. The expression $k(p)$ will denote the shuttle to which the probe p is attached, for p in $\{0, 1, 2, \ldots, 20\}$. For each shuttle we also know its x-offset and y-offset, denoted respectively by O_{p1} and O_{p2}. (Note that O_{p1} and O_{p2} are constants, not variables.) The coordinates of the point where the probe p is attached to the shuttle $k(p)$ are thus equal to $S_{2k(p)} + O_{p1}$ and $S_{2k(p)+1} + O_{p2}$, respectively.

Power chains and avoidance of collisions

In this subsection we first address the problem of modelling the avoidance of collisions above the board, specifically collisions

- between the power chain of a given shuttle and any other shuttle;
- between the power chains of any two shuttles; and
- between two shuttles.

Figure 2 describes the board regions considered when studying collisions. We will first consider modelling the avoidance of collisions between the power chain of a given shuttle and any other shuttle. Figure 3 illustrates the *forbidden region* corresponding to the front-left shuttle for each of five cases (note that Case 2 is illustrated by two subcases), where the forbidden region denotes (roughly) the region that can be occupied by the power chain of the front-left shuttle. Each of the five cases is described precisely below. Our definition of forbidden region is too strict, in the sense that a shuttle or power chain could occupy part of the forbidden region without interfering with the power chain of the shuttle under consideration. A more precise definition, however, would be very difficult to

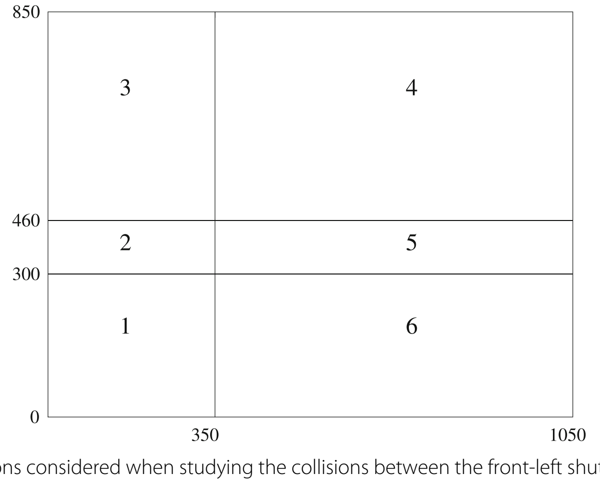

Fig. 2 The six regions considered when studying the collisions between the front-left shuttle and another shuttle

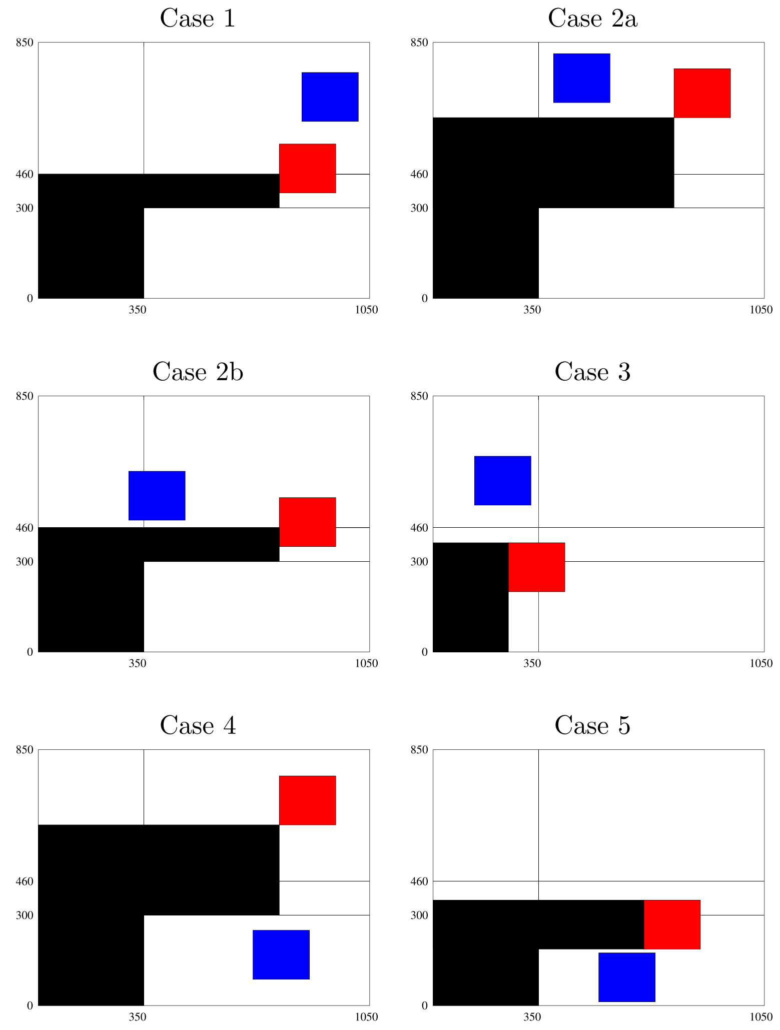

Fig. 3 Illustration of the various cases when studying the collisions between the front-left shuttle and another shuttle : in Case 1 the other shuttle is included within the vertical strip, in Cases 2a, 2b, and 3 within the upper horizontal strip, and in Cases 4 and 5 within the lower horizontal strip (see the text for more details)

model, since a power chain is a flexible object containing several springs and may assume many different positions.

We now give the (mathematical) description of forbidden region that we will use. Let (S_0, S_1) denote the coordinates of the preferred corner of the front-left shuttle. We first define the *large rectangle*: if S_1 is at least 460, the large rectangle denotes the set

$$\{(x,y) \mid 0 \le x \le S_0, 0 \le y \le S_1\};$$

if S_1 is less than 460, it denotes the set

$$\{(x,y) \mid 0 \le x \le S_0, 0 \le y \le \min(S_1 + 160, 460)\}.$$

If S_0 is at most 350, the forbidden region is precisely the large rectangle. If S_0 is greater than 350, the forbidden region is the set difference between the large rectangle and the *small rectangle*, where the latter is defined as

$$\{(x,y) \mid 350 \le x \le S_0, 0 \le y \le \min(S_1, 300)\}.$$

We now turn to the five cases that arise when considering the avoidance of collisions between the power chain of the front-left shuttle and any other shuttle. In Fig. 3 the front-left shuttle is depicted in red, the other shuttle in blue, and the forbidden region in black.

Let S_0' and S_1' denote the coordinates of the *avoidance corner* of the "other" shuttle (which is anonymous for the time being). This avoidance corner is the front-left corner of the other shuttle. Here are the five cases, which are not mutually exclusive:

1. the other shuttle is included within the vertical strip (i.e., $S_0' \geq S_0$ holds);
2. the other shuttle is included within the upper horizontal strip (i.e., $S_1' \geq \max\{460, S_1\}$ holds) and S_1 is at least equal to 300;
3. the other shuttle is included within the upper horizontal strip (i.e., $S_1' \geq S_1 + 160$ holds) and S_1 is at most equal to 300;
4. the other shuttle is included within the lower horizontal strip (i.e., $S_0' \geq 350$ and $S_1' + 160 \leq 300$ hold), S_0 is at least equal to 350, and S_1 at least equal to 300;
5. the other shuttle is included within the lower horizontal strip (i.e., $S_0' \geq 350$ and $S_1' + 160 \leq S_1$ hold), S_0 is at least equal to 350, and S_1 at most equal to 300.

To model these collision avoidance constraints, we define five binary variables y_r (for $r \in \{0, 1, 2, 3, 4\}$), each of which corresponding to one of the cases listed above. The first constraint below means that one of the five cases must be chosen in order to verify that the avoidance corner (i.e., the front-left corner of the "other" shuttle) does not belong to the forbidden region. The second, third, and fourth constraints ensure that if y_{i-1} (for i in $\{1, 2, 3, 4, 5\}$) has the value 1, the conditions on S_0 and S_1 in Case i are indeed verified. The fifth (resp. eighth) constraint ensures that the coordinates S_0' and S_1' satisfy the conditions in Case 1 (resp. 3). Finally the sixth and seventh (resp. ninth and tenth, eleventh and twelfth) constraints ensure that S_0' and S_1' satisfy the conditions in Case 2 (resp. 4, 5).

$$
\begin{aligned}
&y_0 + y_1 + y_2 + y_3 + y_4 = 1 \\
&\qquad S_0 \geq 350(y_3 + y_4) \\
&\qquad S_1 \geq 300(y_1 + y_3) \\
&\qquad S_1 \leq 300(y_2 + y_4) + 850(1 - y_2 - y_4) \\
&\qquad S_0' - S_0 \geq -1050(1 - y_0) \\
&\qquad S_1' - 460 \geq -850(1 - y_1) \\
&\qquad S_1' - S_1 \geq -850(1 - y_1) \\
&\qquad S_1' - S_1 - 160 \geq -850(1 - y_2) \\
&\qquad S_0' \geq 350 y_3 \\
&\qquad S_1' + 160 \leq 300 y_3 + 850(1 - y_3) \\
&\qquad S_0' \geq 350 y_4 \\
&\qquad S_1' + 160 - S_1 \leq 850(1 - y_4)
\end{aligned}
$$

We now simplify these constraints as follows.

$$y_0 + y_1 + y_2 + y_3 + y_4 = 1 \tag{1}$$

$$-S_0 + 350 y_3 + 350 y_4 \leq 0 \tag{2}$$

$$-S_1 + 300 y_1 + 300 y_3 \leq 0 \tag{3}$$

$$S_1 + 550y_2 + 550y_4 \leq 850 \tag{4}$$

$$-S'_0 + S_0 + 1050y_0 \leq 1050 \tag{5}$$

$$-S'_1 + 850y_1 \leq 390 \tag{6}$$

$$-S'_1 + S_1 + 850y_1 \leq 850 \tag{7}$$

$$-S'_1 + S_1 + 850y_2 \leq 690 \tag{8}$$

$$-S'_0 + 350y_3 \leq 0 \tag{9}$$

$$S'_1 + 550y_3 \leq 690 \tag{10}$$

$$-S'_0 + 350y_4 \leq 0 \tag{11}$$

$$S'_1 - S_1 + 850y_4 \leq 690 \tag{12}$$

The above constraints are "abstract" in the sense that S'_0 and S'_1 represent the avoidance corner of an anonymous shuttle. Actually, to model the requirement that there is no collision between the power chain of the front-left shuttle and any other shuttle, we need three groups of 12 constraints, each corresponding to one of the following shuttles: the back-left shuttle ($S'_0 = S_2, S'_1 = S_3 - 160$), the back-right shuttle ($S'_0 = S_4 - 195, S'_1 = S_5 - 160$), and the front-right shuttle ($S'_0 = S_6 - 195, S'_1 = S_7$). We then obtain a complete set of constraints for the front-left shuttle, i.e., 36 constraints. This set requires the introduction of 15 binary variables. We need similar sets of constraints and binary variables for the other shuttles. In total there will be 60 binary variables and 144 constraints.

We must now model the requirement that two avoidance regions cannot overlap, so as to ensure that two power chains cannot "cross." Consider the avoidance region of the front-left shuttle and its relationships with the other avoidance regions.

- Given that no two shuttles intersect (see below) and no shuttle has a non-empty intersection with the avoidance region of any other shuttle, it is enough to impose the condition $S_3 \geq S_1$ in order to prevent the avoidance regions of the front-left and back-left shuttles from having a non-empty intersection.
- The back-right shuttle cannot belong to the lower horizontal region (or at least the part of the lower horizontal region that is not already in the vertical region). Otherwise the avoidance region of the back-right shuttle would have a non-empty intersection with the front-left shuttle **or** the avoidance region of the front-left shuttle. In practice this means that the variables y_8 and y_9 must equal 0.
- The front-right shuttle cannot belong to the upper horizontal region (or at least the part of the upper horizontal region that is not already in the vertical region). Otherwise the avoidance region of the front-right shuttle would have a non-empty intersection with the front-left shuttle **or** the avoidance region of the front-left shuttle. In practice this means that the variables y_{11} and y_{12} must equal 0.

The above conditions will ensure that there is no collision between the power chains in pairs of shuttles that include the front-left shuttle. Similar conditions are needed for the three other pairs of shuttles. Therefore to avoid collisions between power chains (on one hand) and shuttles or other power chains (on the other), we can use the above constraints (after forcing some variables to equal 0) and add the following constraints to the model.

$$S_3 \geq S_1 \tag{13}$$

$$S_5 \geq S_7 \tag{14}$$

Finally we turn to collisions between shuttles. We consider again the front-left shuttle and assume that S'_0 and S'_1 are the coordinates of the front-left corner of the "other" shuttle. The variable y_{60} (resp. y_{61}, y_{62}, y_{63}) takes the value 1 if and only if the "other" shuttle is to the left (resp. above, to the right, below) of the front-left shuttle. The following constraints ensure that the two shuttles will not overlap.

$$\begin{aligned}
y_{60} + y_{61} + y_{62} + y_{63} &= 1\\
S'_0 + 195 &\le S_0 + 1050(1 - y_{60})\\
S'_1 + 850(1 - y_{61}) &\ge S_1 + 160\\
S'_0 + 1050(1 - y_{62}) &\ge S_0 + 195\\
S'_1 + 160 &\le S_1 + 850(1 - y_{63})
\end{aligned}$$

We obtain the following system after simplifying them.

$$\begin{aligned}
y_{60} + y_{61} + y_{62} + y_{63} &= 1 && (15)\\
-S_0 + S'_0 + 1050y_{60} &\le 855 && (16)\\
S_1 - S'_1 + 850y_{61} &\le 690 && (17)\\
S_0 - S'_0 + 1050y_{62} &\le 855 && (18)\\
-S_1 + S'_1 + 850y_{63} &\le 690 && (19)
\end{aligned}$$

As before, the variables S'_0 and S'_1 must be replaced by the correct expressions for every shuttle other than the front-left shuttle. The three other pairs (back-left and back-right, back-left and front-right, back-right and front-right) must also be taken into account. For this kind of constraints a total of 24 binary variables and 30 relations are needed.

We now have a complete system of constraints preventing collisions between shuttles and power chains that are above the board. Of course a similar system is required for the shuttles and power chains that are below the board. All these constraints will be referred to as the *collision avoidance constraints* and denoted by $CAC\ (S_u, y_r)$, where the variables of the form y_r are the *auxiliary variables*.

Probes, points of interest, and reachability constraints

We now introduce the binary variables $X_{\alpha p}$ for every point α and every probe p, where $X_{\alpha p}$ is a binary variable that equals 1 if and only if probe p (attached to the shuttle $k(p)$) can reach the point α (whose coordinates are denoted by $P_{\alpha 1}$ and $P_{\alpha 2}$, respectively). We wish to model the constraint that if $X_{\alpha p}$ equals 1, then the probe p can actually reach the point α. Given the system we are studying, this amounts to saying that if $X_{\alpha p}$ equals 1, the expression $|S_{2k(p)} + O_{p1} - P_{\alpha 1}|$ should be at most 30.0 and the expression $|S_{2k(p)+1} + O_{p2} - P_{\alpha 2}|$ at most 33.5. In other words, if $X_{\alpha p}$ equals 1, the following relations must be satisfied.

$$\begin{aligned}
S_{2k(p)} + O_{p1} - P_{\alpha 1} &\le 30.0\\
-S_{2k(p)} - O_{p1} + P_{\alpha 1} &\le 30.0\\
S_{2k(p)+1} + O_{p2} - P_{\alpha 2} &\le 33.5\\
-S_{2k(p)+1} - O_{p2} + P_{\alpha 2} &\le 33.5
\end{aligned}$$

We can rewrite them as follows.

$$\begin{aligned} S_{2k(p)} &\le -O_{p1} + P_{\alpha 1} + 30.0 \\ -S_{2k(p)} &\le O_{p1} - P_{\alpha 1} + 30.0 \\ S_{2k(p)+1} &\le -O_{p2} + P_{\alpha 2} + 33.5 \\ -S_{2k(p)+1} &\le O_{p2} - P_{\alpha 2} + 33.5 \end{aligned}$$

The variable $S_{2k(p)}$ (resp. $S_{2k(p)+1}$) must satisfy the above constraints if $X_{\alpha p}$ equals 1; otherwise it is bounded from above (resp. below) by its upper (resp. lower) bound. Here are the constraints that we include into the model.

$$S_{2k(p)} \le \left(-O_{p1} + P_{\alpha 1} + 30.0\right) X_{\alpha p} + UB_{2k(p)} \left(1.0 - X_{\alpha p}\right) \tag{20}$$

$$-S_{2k(p)} \le \left(O_{p1} - P_{\alpha 1} + 30.0\right) X_{\alpha p} - LB_{2k(p)} \left(1.0 - X_{\alpha p}\right) \tag{21}$$

$$S_{2k(p)+1} \le \left(-O_{p2} + P_{\alpha 2} + 33.5\right) X_{\alpha p} + UB_{2k(p)+1} \left(1.0 - X_{\alpha p}\right) \tag{22}$$

$$-S_{2k(p)+1} \le \left(O_{p2} - P_{\alpha 2} + 33.5\right) X_{\alpha p} - LB_{2k(p)+1} \left(1.0 - X_{\alpha p}\right) \tag{23}$$

These inequalities (called *reachability constraints*) are required for each couple (α, p). The set of reachability constraints will be denoted by $REACH\left(S_u, X_{\alpha p}\right)$.

Incompatibility constraints

These constraints are not necessary for modelling our problem but allow us to strengthen its linear programming relaxation. Assume that α and β denote two points of interest and p and q two probes attached to the same shuttle. It is easy to derive from the reachability constraints some conditions implying that $X_{\alpha p}$ and $X_{\beta q}$ cannot both equal 1. So for each pair of couples (α, p) and (β, q) satisfying those conditions, we may add the constraint $X_{\alpha p} + X_{\beta q} \le 1$ to the model. In practice we do not add all the incompatibility constraints to the model: rather we solve its linear relaxation and add to the model all such constraints that are violated in the current optimal solution and verify $X_{\alpha p} \ge 0.2$ and $X_{\beta q} \ge 0.2$ in the current optimal solution. The set of all incompatibility constraints included into the model will be denoted by $INC\left(X_{\alpha p}\right)$.

Tests and feasibility constraints

Given the current configuration (i.e., a set of values for the variables S_u representing the shuttles coordinates), we wish to determine whether a specific test can be carried out or not. We introduce, for each ℓ, a binary variable Y_ℓ that may take the value 1 only if the test ℓ can be carried out within the current configuration. We must find a way of relating the value of Y_ℓ to those of the $X_{\alpha p}$. First we show that the test ℓ can be performed within the current configuration if and only if there is a matching of a certain type in a bipartite graph (denoted G_ℓ) that we now describe. Let N_ℓ denote the set of nets involved in test ℓ and U_ℓ the set of points belonging to some net in N_ℓ. The graph G_ℓ contains two kinds of vertices: the points in U_ℓ and the probes. It contains an edge between point α and probe p if and only if probe p can be used to touch point α within the test ℓ (i.e., the couple (α, p) is *admissible*) and $X_{\alpha p}$ equals 1. Recall that the admissibility of the couple (α, p) depends upon ℓ. We denote by E_ℓ the set of admissible couples for test ℓ. In Fig. 4 all the edges in E_ℓ are represented: the green edges are those defining graph G_ℓ and the black edges represent couples (α, p) that are admissible but verify $X_{\alpha p} = 0$. Note that Net 0 (resp. Net 1, Net 2) is the set $\{0, 1, 2\}$ (resp. $\{3, 4, 5\}$, $\{6, 7, 8\}$).

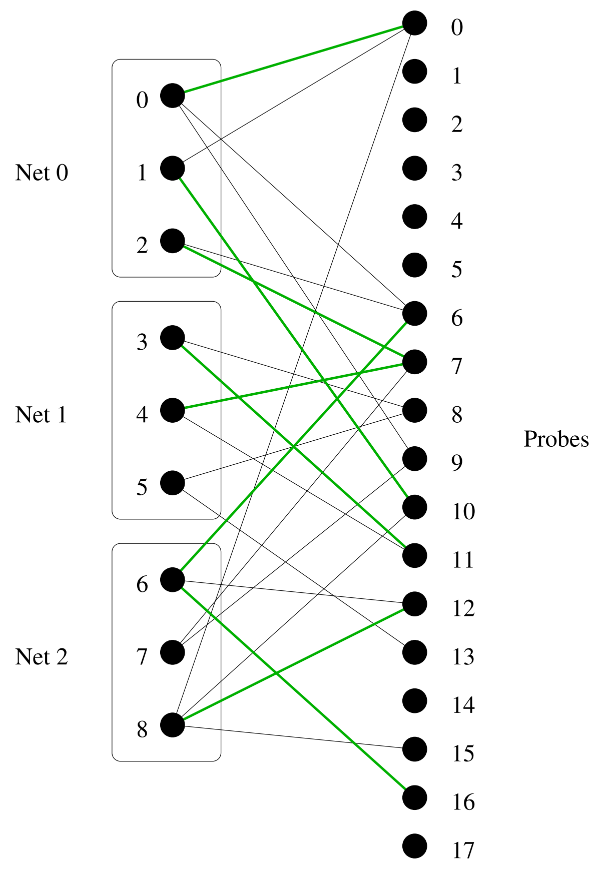

Fig. 4 The graph for a specific test (assuming the number of probes equals 18): the set of all the edges represented is E_ℓ while the green edges are those defining the graph G_ℓ

From the definition of "carrying out a test," it follows that a test can be performed within the current configuration if and only if G_ℓ contains a matching saturating exactly one point within each net in N_ℓ. For instance the green graph G_ℓ in Fig. 4 contains the matching $\{(1,10),(4,7),(8,12)\}$, showing that probe 10 (resp. 7, 12) can touch point 1 (resp. 4, 8) in Net 0 (resp. 1, 2). Hence the test ℓ can be carried out within the current configuration. We now observe that finding a matching saturating exactly one point within each net in N_ℓ is equivalent to finding a system of distinct representatives for the collection of sets $\{Z_i\}$, where Z_i is the set of probes that can be used to touch a point in the net i, i.e.,

$$Z_i = \left\{p \in P \mid (\alpha, p) \text{ is admissible and } X_{\alpha p} = 1 \text{ for some } \alpha \text{ in net } i\right\}$$

and P denotes the set of probes. A classical theorem in combinatorial optimization asserts that the collection of sets $\{Z_i\}_{i \in N_\ell}$ has a system of distinct representatives if and only if the condition

$$\left|\bigcup_{i \in M} Z_i\right| \geq |M|$$

is satisfied for every subset M of N_ℓ (see in particular the seminal text by Ford and Fulkerson [9] and the book by Chvátal [4]). This theorem can be rephrased as follows:

the collection $\{Z_i\}_{i\in N_\ell}$ has a system of distinct representatives if and only if the condition

$$\left(\bigcup_{i\in M} Z_i\right) \cap Q \neq \emptyset$$

holds for every set Q verifying $|\overline{Q}| = |M| - 1$ (where $\overline{Q}$ denotes the complement of Q).

Let M denote any nonempty subset of N_ℓ and $U(M)$ the union of all the points belonging to nets in M. The statement

$$\left(\bigcup_{i\in M} Z_i\right) \cap Q \neq \emptyset$$

is equivalent to the statement

$$\sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in U(M), p\in Q}} X_{\alpha p} \geq 1,$$

by the definition of the Z_i. We thus obtain the following proposition.

Proposition 2.1 *The graph G_ℓ contains a matching saturating exactly one point within each net in N_ℓ if and only if the inequalities*

$$\sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in U(M), p\in Q}} X_{\alpha p} \geq 1$$

hold for all sets M, Q such that M is not empty, Q is a subset of probes, and $|\overline{Q}|$ equals $|M| - 1$ (where $\overline{Q}$ denotes the complement of Q).

For each test ℓ we include into our model the constraints

$$\sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in U(M), p\in Q}} X_{\alpha p} \geq Y_\ell \tag{24}$$

for all sets M, Q such that M is not empty, Q is a subset of probes, and $|\overline{Q}|$ equals $|M| - 1$. They will be called *feasibility constraints* and denoted by *FEAS* $\left(X_{\alpha p}, Y_\ell\right)$. Proposition 2.1 implies that if Y_ℓ equals 1, then the graph G_ℓ contains a matching of the required type and test ℓ can be carried out within the current configuration.

For an illustration we give the feasibility constraints for the test represented in Fig. 4. First we assume that $|M|$ equals 1 and write the following constraints.

$$X_{0,0} + X_{0,6} + X_{0,9} + X_{1,0} + X_{1,10} + X_{2,6} + X_{2,7} \geq Y_\ell$$
$$X_{3,8} + X_{3,11} + X_{4,7} + X_{4,11} + X_{5,8} + X_{5,13} \geq Y_\ell$$
$$X_{6,6} + X_{6,12} + X_{6,16} + X_{7,7} + X_{7,9} + X_{8,0} + X_{8,10} + X_{8,12} + X_{8,15} \geq Y_\ell$$

For the case where $|M|$ equals 2, we need all the constraints of the form

$$\sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in U(M), p\neq q}} X_{\alpha p} \geq Y_\ell$$

for some probe q. For instance, if M consists of nets 0 and 1 and q equals 6, we have

$$X_{0,0} + X_{0,9} + X_{1,0} + X_{1,10} + X_{2,7} + X_{3,8} + X_{3,11} + X_{4,7} + X_{4,11} + X_{5,8} + X_{5,13} \geq Y_\ell.$$

Finally, for the case where $|M|$ equals 3, we need all the constraints of the form

$$\sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in U(M),p\neq q,q'}} X_{\alpha p} \geq Y_\ell$$

for some pair of probes $\{q, q'\}$. For instance, if q equals 6 and q' equals 7, we have

$$X_{0,0} + X_{0,9} + X_{1,0} + X_{1,10} + X_{3,8} + X_{3,11} + X_{4,11} + X_{5,8} + X_{5,13} + X_{6,12} + X_{6,16} + X_{7,9} + X_{8,0} + X_{8,10} + X_{8,12} + X_{8,15} \geq Y_\ell.$$

When the model containing these feasibility constraints returns a solution, one must do a little bit of work to compute the couples (α, p) that describe the implementation of test ℓ (for each ℓ). Actually it suffices to solve a bipartite matching problem in the graph G_ℓ corresponding to this solution. We now comment on the number of feasibility constraints for a given ℓ. Assuming that there are 21 probes (as in the case of our data sets), this number equals

$$\sum_{m=1}^{|N_\ell|} \binom{|N_\ell)|}{m} \sum_{Q:|Q|=21-(m-1)} 1 = \sum_{m=1}^{|N_\ell|} \binom{|N_\ell|}{m} \binom{21}{m-1}$$

and grows rapidly as a function of $|N_\ell|$. This is not an issue because in our data sets $|N_\ell|$ is almost always less than five. Nonetheless one might ask why we do not model directly the requirement that there be a matching between nets and probes (in order to carry out a specific test). The reason is that to do so, we would need to introduce variables of the form $X_{\alpha p\ell}$ and the resulting model would contain too many variables.

If one does not want to use the above feasibility constraints when $|N_\ell|$ is at least three (note that the number of constraints equals 276 when N_ℓ contains three nets!), an alternative is to introduce the supplementary variables $X_{\alpha p\ell}$ and the following constraints.

$$\sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in U(N_\ell)}} X_{\alpha p\ell} \leq 1 \text{ for all } p \in P$$

$$Y_\ell \leq \sum_{\substack{(\alpha,p)\in E_\ell \\ \alpha\in k,p\in P}} X_{\alpha p\ell} \leq 1 \text{ for all } k \in N_\ell$$

$$X_{\alpha p\ell} \leq X_{\alpha p} \text{ for all } (\alpha, p, \ell)$$

The partial covering problem

It is natural to ask the following question: what is the maximum number of tests that can be carried out while the system is in a given configuration? This problem, which we call the *partial covering problem*, can be formulated as follows: maximize the function $\sum_\ell Y_\ell$ subject to the collision avoidance constraints $CAC(S_u, y_r)$, the reachability constraints $REACH(S_u, X_{\alpha p})$, the incompatibility constraints $INC(X_{\alpha p})$, the feasibility constraints $FEAS(X_{\alpha p}, Y_\ell)$, the lower and upper bound constraints on all the variables, and the integrality constraints on the binary variables. The resulting mixed integer linear program can be solved by using a commercial solver.

A greedy algorithm for the covering problem

The covering problem can be solved in a heuristic fashion through repeated calls of an algorithm for solving the partial covering problem. This method is summarized in Algorithm 1. Note that in the first line of the loop body, $\mathcal{T}$ can be replaced by a subset of tests. Then one must determine which of the omitted tests are covered by the configuration C before removing them from $\mathcal{T}$.

Algorithm 1 A greedy algorithm for the covering problem

```
1: $\mathcal{C} \longleftarrow$ {the initial configuration}
2: let $\mathcal{T}$ denote the set of all tests
3: remove from $\mathcal{T}$ the tests that can be carried out within the initial configuration
4: $n \longleftarrow 1$
5: while $\mathcal{T}$ is not empty do
6:     compute a configuration $C$ covering the maximum number of tests in $\mathcal{T}$
7:     include $C$ in $\mathcal{C}$
8:     remove from $\mathcal{T}$ the tests covered by $C$
9:     $n \longleftarrow n + 1$
10: end while
```

The optimal sequence problem

Let $\mathcal{C}$ be the set of configurations computed by the above algorithm and n its cardinality. Thus $\mathcal{C}$ is of the form $\{C_1, C_2, \ldots, C_n\}$, where C_i denotes the ith configuration (C_1 being the initial configuration). Let also $d\left(C_i, C_j\right)$ denote the time taken by the FP system to go from the configuration C_i to the configuration C_j. We wish to compute the best sequence of configurations, i.e., a sequence $\left(C_1 = C_{\sigma(1)}, C_{\sigma(2)}, \ldots, C_{\sigma(n)}, C_1\right)$ (where σ denotes a permutation) that minimizes

$$\sum \{d\left(C_i, C_j\right) \mid j \text{ follows } i \text{ in the sequence}\}.$$

When a shuttle moves from one configuration to the next, it travels at the same speed in the horizontal direction as in the vertical direction, and thus the time it takes to do so is proportional to the L_∞ distance between the respective positions of the shuttle in the two configurations. Hence we assume that $d\left(C_i, C_j\right)$ is the L_∞ distance between C_i and C_j.

Since the distances are symmetric (meaning that $d\left(C_i, C_j\right) = d\left(C_j, C_i\right)$ holds for any pair $\{i, j\}$), we can model the optimal sequence problem as a Travelling Salesman Problem (TSP) on an undirected graph. The TSP itself can be modelled by using *degree constraints* (each vertex must be of degree 2) and *subtour elimination constraints* (enforcing the condition that there be at least two edges between a subset S of vertices and its complement, for any S verifying $2 \leq |S| \leq n - 2$). We introduce the binary variables z_{ij} for $1 \leq i < j \leq n$, where z_{ij} equals 1 if and only if the edge ij belongs to the tour. Here is the integer programming formulation of the optimal sequence problem.

$$\min \quad \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d\left(C_i, C_j\right) z_{ij}$$

subject to

$$\sum_{j=1}^{i-1} z_{ji} + \sum_{j=i+1}^{n} z_{ij} = 2 \qquad \forall i \in \{1,2,\ldots,n\} \tag{25}$$

$$\sum_{i\in S}\sum_{\substack{j\notin S\\ j>i}} z_{ij} + \sum_{i\in S}\sum_{\substack{j\notin S\\ j<i}} z_{ji} \geq 2 \qquad \forall S \subset \{1,2,\ldots,n\}, 2 \leq |S| \leq n-2 \tag{26}$$

$$z_{ij} \in \{0,1\} \qquad \forall i, \forall j \text{ such that } i<j$$

A global formulation of the planning problem

We now give a formulation of the planning problem that is global but unlikely to be solved for large data sets. In this formulation we are looking for several configurations at once, so that each test is covered by at least one configuration. We assume that there are at most n configurations. We introduce a group of variables $\left(S^i_u, y^i_r, X^i_{\alpha p}, Y^i_\ell\right)$ for each $i \in \{1,2,\ldots,n\}$, that is, for each potential configuration. The variable Y^i_ℓ equals 1 if and only if the test ℓ is covered by the configuration represented by the shuttles coordinates S^i_u. For each i the vector $\left(S^i_u, y^i_r, X^i_{\alpha p}, Y^i_\ell\right)$ must satisfy the constraints in $CAC\left(S^i_u, y^i_r\right)$, $REACH\left(S^i_u, X^i_{\alpha p}\right)$, $INC\left(X^i_{\alpha p}\right)$, and $FEAS\left(X^i_{\alpha p}, Y^i_\ell\right)$, as well as the bound and nonnegativity constraints on the variables and the integrality constraints on all the variables except the S^i_u.

To these constraints one must add constraints ensuring that each test is covered by at least one configuration. These constraints can be expressed as $\sum_{i=1}^n Y^i_\ell \geq 1$ for every ℓ. We now introduce variables and constraints to model the choice of the optimal configuration sequence. For i in $\{1,2,\ldots,n\}$, let w_i be a binary variable that equals 1 if and only if Configuration i is chosen. For any nonempty proper subset S of $\{1,2,\ldots,n\}$, let w_S be a binary variable that equals 1 if and only if S contains at least one of the chosen configurations. As in the subsection "The optimal sequence problem", we introduce the variables z_{ij} defined as follows: z_{ij} (for $1 \leq i < j \leq n$) equals 1 if and only if Configuration i appears immediately before or after Configuration j in the sequence of configurations. Finally we let d_{ij} denote the L_∞ distance between Configurations i and j. We can now propose a model for choosing an optimal configuration sequence.

$$\min \quad \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} d_{ij} z_{ij}$$

subject to

$$\sum_{i=1}^{n} Y^i_\ell \geq 1 \qquad \forall \ell \tag{27}$$

$$Y^i_\ell \leq w_i \qquad \forall i \in \{1,2,\ldots,n\}, \forall \ell \tag{28}$$

$$w_i \leq w_S \qquad \forall i \in S, \forall S \subset \{1,2,\ldots,n\}, S \neq \emptyset \tag{29}$$

$$\sum_{j=1}^{i-1} z_{ji} + \sum_{j=i+1}^{n} z_{ij} = 2w_i \qquad \forall i \in \{1,2,\ldots,n\} \tag{30}$$

$$\sum_{i\in S}\sum_{\substack{j\notin S\\ j>i}} z_{ij} + \sum_{i\in S}\sum_{\substack{j\notin S\\ j<i}} z_{ji} \geq 2(w_S + w_{\overline{S}} - 1) \qquad \forall S \subset \{1,2,\ldots,n\}, S \neq \emptyset \tag{31}$$

$$\begin{aligned}
d_{ij} &\geq |S^i_u - S^j_u| && \forall u, \forall i, \forall j \text{ such that } i < j \qquad (32)\\
CAC\left(S^i_u, y^i_r\right) &&& \forall i\\
REACH\left(S^i_u, X^i_{\alpha p}\right) &&& \forall i\\
INC\left(X^i_{\alpha p}\right) &&& \forall i\\
FEAS\left(X^i_{\alpha p}, Y^i_\ell\right) &&& \forall i\\
Y^i_\ell &\in \{0,1\} && \forall i \in \{1,2,\dots,n\}, \forall \ell\\
w_i &\in \{0,1\} && \forall i \in \{1,2,\dots,n\}\\
w_S &\in \{0,1\} && \forall S \subset \{1,2,\dots,n\}, S \neq \emptyset\\
z_{ij} &\in \{0,1\} && \forall i, \forall j \text{ such that } i < j\\
d_{ij} &\geq 0 && \forall i, \forall j \text{ such that } i < j\\
S^i_u &\geq 0 && \forall i, \forall u\\
y^i_r &\in \{0,1\} && \forall i, \forall r\\
X^i_{\alpha p} &\in \{0,1\} && \forall i, \forall (\alpha, p)
\end{aligned}$$

This model is a mixed integer program with linear constraints and a nonlinear objective function. It can be linearized by introducing the nonnegative real variables D_{ij}, where D_{ij} stands for the product $d_{ij}z_{ij}$ for each i and j with $i < j$. After replacing the objective function by $\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} D_{ij}$ and adding the constraints $D_{ij} \geq d_{ij} - M(1 - z_{ij})$ (where M is a constant larger than any distance between two configurations), we obtain a model of our problem that is actually a mixed integer linear program.

We expect that n (the maximum number of configurations) will be relatively small for most data sets, i.e., smaller than 30. If the number of tests is large, however, the above MILP will be hard to solve and one will have to investigate the use of column generation or other decomposition techniques for solving our problem. We also note that the set of solutions of this model exhibits many symmetries. This difficulty could be partially overcome by introducing the constraints $w_i \geq w_{i+1}$ (for i in $\{1, 2, \dots, n-1\}$) into the model.

Results and discussion

In this section we present results pertaining to the solution of the covering and sequencing subproblems. As mentioned in the subsection "A global formulation of the planning problem", the global formulation cannot be used for large data sets, particularly the kind of data sets that the Company must handle. We start by describing the format of each data set, consisting of a *probe file*, a *point file*, and a *test file*. The probe file contains the description of each probe, namely: a numerical identifier; the string "top" or "bot" to indicate whether the shuttle to which the probe is attached is on top or on bottom of the board; a string indicating the position of that shuttle (for instance "fl" refers to a front-left shuttle); and the coordinates of the point of attachment of the probe with respect to the preferred corner of the shuttle to which it is attached. Here is an example of such a description.

```
0 top fl 65.0 79.0
```

This means that Probe 0 is attached to the front-left shuttle on top of the board and that it is attached to this shuttle at the point of coordinates $(65.0, 79.0)$ (since the preferred corner of a front-left shuttle is the point of coordinates $(0.0, 0.0)$). The point file contains the description of each point, i.e., a numerical identifier and the coordinates of that point. For example the line corresponding to Point 0 is the following.

```
0 525.0 425.0
```

The test file consists of a sequence of test descriptions (each of which occupying several lines). For instance the description of Test 4908 is as follows.

```
test 4908 2
net 9816 3
3120
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
3121
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
3122
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
net 9817 4
5268
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
5269
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
5270
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
5271
1 2 3 4 5 6 7 8 10 11 13 14 15 16 19 20
```

Note that the second number on the first line represents the number of nets within the test (2, in this example). Therefore the first line in the test description is followed by the description of the first net, Net 9816, followed by the description of the second one, Net 9817. Net 9816 consists of three points, represented by their identifiers and whose coordinates are stored in the point file. The points in question are Points 3120, 3121, and 3122, and each point identifier is followed (on the next line) by the list of probes that may touch that particular point. Naturally Net 9817 is described in a similar fashion. Table 1 summarizes the characteristics of the four instances that we used for our tests. It turns out that some of the tests in each of the four data sets are infeasible, i.e., there are tests that no configuration can "cover." For this reason our program begins by determining which tests are infeasible through solving a series of partial covering problems (each with a data set consisting of one test). Then we solve the partial covering problem (see section "Methods") for the remaining tests, each of which is feasible. Note that we did not include into the model the tests that consist of 5 nets or more, because the corresponding feasibility constraints (see the subsection "Tests and feasibility constraints") are too numerous. In any case only Dataset 3 contains such tests and there are only 6 of them.

In our experiments we solved the covering problem by using Cplex to find a solution of each of the partial covering problems. Note that Cplex implements a branch-and-bound

Table 1 Characteristics of the instances

Instance identifier	Nb. of probes	Nb. of points	Nb. of tests	Nb. of nets
Dataset3	21	1034	2035	4388
Dataset4	21	7445	9832	19664
7700FC	21	3016	4423	10996
pinPCB_15mils	21	6659	3331	6659

algorithm for solving mixed integer linear programming problems (see Nemhauser and Wolsey [16] for a description of the branch-and-bound method). Our initial intention was to solve the covering problem by solving partial covering problems until all the tests had been covered, as explained in the subsection "A greedy algorithm for the covering problem". Unfortunately the number of constraints of our model is huge and including into it all the feasible tests consumed too much memory. Therefore instead of solving a partial covering problem including all the feasible tests, we included at most 400 tests in each of the successive partial covering problems solved by our algorithm. The configuration obtained after solving a given partial covering problem may actually cover more than the number of covered tests indicated in the solution; therefore we look for all the tests covered by the current configuration before solving the next partial covering problem.

The mixed integer linear program (MILP) corresponding to a partial covering problem has a weak linear relaxation and many of its solutions have the same objective function value. Hence finding a (provably) optimal solution of a given partial covering problem takes too much time and we decided to put a limit on its execution time. For each of the four data sets, we tried three such limits: 90 seconds, 180 seconds, and 270 seconds, respectively. For instance, in Table 2, the identifier 7700FC-180 refers to an experiment with data set 7700FC in which we let the Cplex MILP algorithm run for at most 180 seconds when attempting to solve a partial covering problem. We ran our tests on a machine with an Intel Core i7-860, 2.80GHz, and a 2G memory.

Intuitively, if we spend more time finding a good solution for each of the partial covering problems, we should be able to cover all the tests with fewer configurations and hence obtain a better solution of the planning problem. With one exception (Dataset4-90), the results in Table 2 show that spending 270 seconds to solve each partial covering problem always yields a solution that is better (i.e., of smaller length) than those obtained after spending only 90 or 180 seconds on each partial covering problem. This is true even if spending 90 or 180 seconds on each partial covering problem yields a solution with fewer configurations. Overall the results are very encouraging, because the system currently in use at the Company produces plans that may include hundreds of configurations while

Table 2 Results. *Real time* represents the time (in minutes) needed to solve the covering problem, *User time* the time (in minutes) consumed by all the processors, *Nb. of config.* the number of partial covering problems that were solved, and *Sol. value* the optimal value of the sequencing problem described in the subsection "The optimal sequence problem"

Instance	Nb. feasible tests	Real time	User time	Nb. of config.	Sol. value
Dataset3-90	1404	65	192	28	12531.5
Dataset3-180	1404	80	281	24	12927.5
Dataset3-270	1404	116	387	26	11370.3
Dataset4-90	9813	57	141	17	6887.0
Dataset4-180	9813	62	188	14	7616.6
Dataset4-270	9813	81	292	15	7016.7
7700FC-90	2855	89	139	20	9953.6
7700FC-180	2855	104	185	17	8347.6
7700FC-270	2855	77	206	15	6844.7
pinPCB_15mils-90	2360	60	103	13	6178.8
pinPCB_15mils-180	2360	65	164	11	5301.2
pinPCB_15mils-270	2360	81	261	11	5284.0

our solutions include fewer than 30 configurations. The real time consumed for computing a solution is considered reasonable by the Company, since the time needed to test a single board is itself substantial. Therefore it is feasible to spend one hour or so preparing the sequence of configurations that will be used for testing a specific board.

Conclusion

In this article we have shown that the use of modelling and mathematical programming yields solutions of the test planning problem that are much better than those currently used by the Company. The new solutions can be obtained within times that are reasonable. Our results indicate that the system currently used by the Company could be replaced by a system based on mathematical programming. More work remains to be done, however, in order to strenghten the formulation of the partial covering problem. In particular we would like to find new families of cutting planes, whose introduction into the model will speed up the solution process and thus yield better solutions of the partial covering problem. Another avenue is to design heuristics (not making use of MILP models) in order to cover a large number of tests with few configurations. One could then use the MILP model described in section "Methods" to cover the remaining tests, presumably those complicated tests that include three or more nets. These improvements may enable us to find solutions of the planning problem that are as good as those reported in this article but take less time to compute. As for the model in the subsection "A global formulation of the planning problem", we could test it on small data sets and then look for more sophisticated approaches for solving it, in the hope of computing solutions for real-world problems within reasonable times.

Funding
Some of the work described in this article was carried out while Mario Morfin was a postdoctoral fellow at the Company. His two internships were supported by Mitacs Inc. and NSERC, respectively, and were also partially supported by the Company.

Authors' contributions
The four authors started collaborating with the Company during the 2010 Fields-MITACS Industrial Problem-Solving Workshop. Together they proposed a first attempt at modelling the problem submitted by the Company to the workshop. MM then spent two semesters at the Company in order to understand and improve the system the Company is currently using. His work was supervised by SC. MM and OM worked on the model presented in our article and OM completed the computational work and the article itself. All authors read and approved the final manuscript.

Competing interests
None of the authors has any competing interests (financial or otherwise) regarding the work described in this article.

Author details
[1]School of Information Technology, York University, 3068 TEL Building, 4700 Keele Street, Toronto M3J 1P3, Canada. [2]GERAD, HEC Montréal and Département d'informatique, UQAM, 3000 Côte-Sainte-Catherine, Montréal H3T 2A7, Canada. [3]MOAI Solutions Inc., 3-143 Arlington Ave., Toronto M6C 2Z3, Canada. [4]Department of mathematics, University of Toronto, 40 St. George St, room 6290, Toronto M5S 2E4, Canada.

Received: 15 September 2016 Accepted: 11 August 2017
Published online: 20 September 2017

References
1. Ahn H-R, Bae SW, Demaine ED, Demaine ML, Kim S-S, Korman M, Reinbacher I, Son W (2011) Covering points by disjoint boxes with outliers. Comput Geometry 44(3):178–90
2. Applegate DL, Bixby RE, Chvátal V, Cook WJ (2006) The Traveling Salesman Problem: a computational study. Princeton Series in Applied Mathematics. Princeton University Press, Princeton
3. Arkin EM, Hassin R (1994) Approximation algorithms for the geometric covering salesman problem. Discret Appl Math 55(3):197–218

4. Chvátal V (1983) Linear programming. Freeman and Company, WH. New York
5. Current JR, Schilling DA (1989) The Covering Salesman Problem. Transp Sci 23(3):208–13
6. Dimitrijevic V, Saric Z (1997) An efficient transformation of the generalized traveling salesman problem into traveling salesman problem on digraphs. Inf Sci 102:105–10
7. Fischetti M, Gonzales JJS, Toth P (1995) The symmetric generalized traveling salesman polytope. Networks 26:113–23
8. Fischetti M, Gonzales JJS, Toth P (1997) A Branch-and-cut Algorithm for the Symmetric Generalized Traveling Salesman Problem. Oper Res 45:378–94
9. Ford LR, Fulkerson DR (1962) Flows in Networks. Princeton University Press, Princeton
10. Kara I, Guden H, Koc ON (2012) New formulations for the generalized traveling salesman problem. In: Proceedings of the 6th International Conference on Applied Mathematics, Simulation, Modelling World Scientific and Engineering Academy and Society (WSEAS). Stevens Point, Wisconsin. pp 60–5
11. Laporte G, Palekar U (2002) Some applications of the clustered travelling salesman problem. J Oper Res Soc 53:972–6
12. Laporte G, Asef-Vaziri A, Sriskandarajah C (1996) Some applications of the generalized traveling salesman problem. J Oper Res Soc 47:1461–7
13. Laporte G, Mercure H, Nobert Y (1987) Generalized travelling salesman problem through n sets of nodes: the asymmetrical cases. Discret Appl Math 18:185–97
14. Laporte G, Nobert Y (1983) Generalized travelling salesman problem through n sets of nodes: an integer programming approach. INFOR 21:61–75
15. Laporte G, Semet F (1999) Computational evaluation of a transformation procedure for the symmetric generalized traveling salesman problem. INFOR 37:114–20
16. Nemhauser GL, Wolsey LA (1988) Integer and combinatorial optimization. Wiley, New York
17. Noon CE, Bean JC (1991) A Lagrangian based approach for the asymmetric generalized traveling salesman problem. Oper Res 39:623–32
18. Pop, PC (2007) New integer programming formulations of the generalized traveling salesman problem. Am J Appl Sci 4(11):932–7
19. Riordan T (2011) Solving the problem of the traveling politician. https://eqn.princeton.edu/2011/12/_text_goes_here/. Accessed 2 Feb 2017