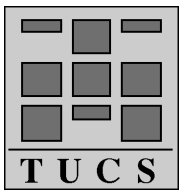
Estimating the Production Times in PCB Assembly

Tero Laakso
Mika Johnsson
Tommi Johtela
Jouni Smed
Olli Nevalainen



Turku Centre for Computer Science
TUCS Technical Report No 392
February 2001
ISBN 952-12-0783-3
ISSN 1239-1891

Abstract

Several control problems of PCB assembly industry involve the estimation of the component placement times for various different placement robots. A standard approach, widely used in scientific literature, is to let the time to be a linear function of the number of components. In this paper, we demonstrate that a more realistic model should include additional parameters (e.g., the number of different component types and the board size). Linear multiregression indicates that the coefficient of determination for the improved model is over 90 percent.

Keywords: printed circuit board assembly, placement machines, placement time, estimation of running times

TUCS Research Group
Algorithmics

1 Introduction

Automated placement of electronics components on printed circuit boards (PCBs) is carried out by various different types of placement machines. Each machine type can operate a certain set of component types. The components have different characteristics (e.g., size) and this is reflected in the design of the placement machines. Different mechanical solutions lead to different nominal operation speed, and the same machine can be run at different speeds according to the operated components types. Moreover, the printing time of a PCB depends strongly on the order of component placements.

Generally speaking, the problems encountered in PCB assembly can be divided into four major classes according to the number of different PCBs and machines present in the problem [3, 6]: (1) *single machine optimization* problems (including feeder arrangement, placement sequencing, nozzle assignment, and component retrieval problems), (2) *setup strategies for a single machine* (e.g., unique setup, minimum setup, and group setup strategies), (3) *component allocation to multiple machines* (e.g., balancing the workload of the machines in the same production line), and (4) *scheduling problems* (e.g., job allocation and line sequencing). For an extensive survey on the research, see [6].

This simplified description of the PCB assembly process stresses the view that the whole production capacity should be utilized efficiently. This strive has been the impetus behind a rich research activity in the PCB assembly problems. Typically, the problem formulations regard the real-world situation too complex for a one-level model but rather use a hierarchy of models [2]. The problems are connected to each other so that the solving of the complex problems requires the solutions of the simpler ones. Since the single machine optimization is at the lowest level of the problem hierarchy, it has to be solved each time when solving any of the higher level problems.

In a line balancing system, the aim is to optimize the operation of the machines so that the amount of PCBs in a time unit (i.e., throughput) is maximized. This problem is usually tackled by maximizing the throughput of the bottleneck machine (i.e., the machine which restricts the overall productivity of the line). Line Engineering Package (LEP) of Trilogy 500 by Valor Computerized Systems Ltd. is an efficient line balancer, where the user can create the lines of desired topology and allocate machines from a large collection of machine types, see Figure 1.

Currently, LEP uses two different ways of calculating the PCB production times. On a coarse level, the production time is calculated from the nominal *component time* (i.e., the production time is the sum of a fixed constant for start and finish of the assembly task, and a term which is a product of the

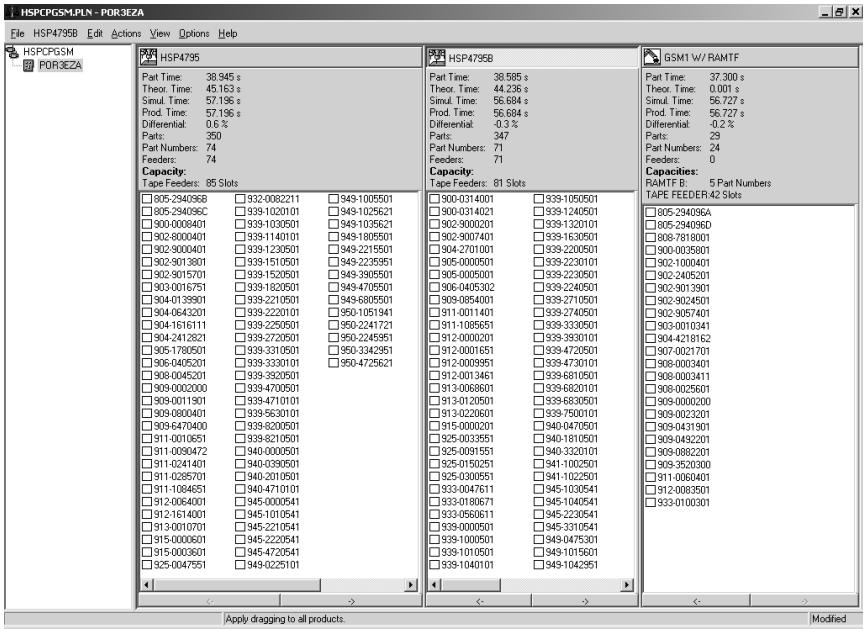


Figure 1: A screenshot of the Line Engineering Package

number of components and the placement time per component). Nowadays, this type of estimation is widely used in commercial balancing systems. The benefit of this method is simplicity and, consequently, high speed. On the other hand, its accuracy is modest and it suits only for finding an initial approximative solution to the balancing problem. When a control software is implemented and tested in real-world production, it becomes soon clear that the above traditional time estimate is not accurate enough. The optimization may succeed to yield a low value for the objective function but the observed real-world production times can be far from optimal.

To get a more detailed solution, LEP uses the *simulation times*, which are obtained from built-in simulators for each machine type. The simulators use, among other things, information about the placement sequence, operation sequence, machine timings, component coordinates, and machine geometry. The best simulators are accurate: according to our experience, the usual error rate of MEP is less than two percent for turret machine types. On the downside, a simulator does complex computations which take a substantial running time. This limits its use in multi-product balancing and production scheduling.

A variant of this method has turned out to be useful when the feeder movements are restricted to be unidirectional [1, 4]. In this method, all components of a given type are collected to a cluster. Two clusters are neighbors

if their components reside in subsequent feeder slots and the components of two neighboring clusters can be installed without any feeder setup costs (i.e., the setup time consists of the xy-table movements only).

These considerations lead to two questions: How accurate are the simple component time estimators and is it possible to construct fast and more accurate estimators?

In this paper, we exemplify the difficulties of estimating the component placement time and consider linear multiregression models for the time estimates. Our results are obtained from a high-speed turret-type placement machine but other machine types manifest similar properties and problems in the time estimation. We begin in Section 2 with a short description of the particular machine type in question. The modeling of the time for performing the placement of n components on a single PCB is discussed in Section 3. The concluding remarks appear in Section 4.

2 A High-Speed Rotary Turret Machine

Let us consider a turret-type placement machine [5], see Figure 2. The discussion and all tests presented in this paper refer to Universal HSP 4795/96 (or Sanyo TCM 3000) printing machines. The main body of the machine includes a transport line, a holding table for the PCBs and a rotary turret placement head for holding the vacuum nozzles. The feeder unit moves horizontally and comprises the required component reels. The number of components reels in the feeder unit is limited and each reel occupies a certain number of storage places (called feeder slots). This number depends on the width of the part. The task of the feeder unit is to bring the proper reel to the so-called pick-up position where the placement head picks up a component with the appropriate vacuum nozzle. After that, the turret rotates until the nozzle is at the printing position (which is 180° from the pick-up position). The intermediate positions are for centering, inspection, and rotation of the component. Because the rotary turret contains several placement heads, there are several components ready in the turret and the feeder movement is performed a given number of turret rotation steps prior to the actual placement operation. Because the printing position is fixed, the table holding the PCB must be moved by two independent step-motors to the proper location.

The turret rotates on steps, and each step lasts a fixed time slice during which the feeder unit and the holding table can move for free (i.e., their movements are simultaneous to the rotation step and do not increase the total time for the assembly task). Mathematically speaking, we have the

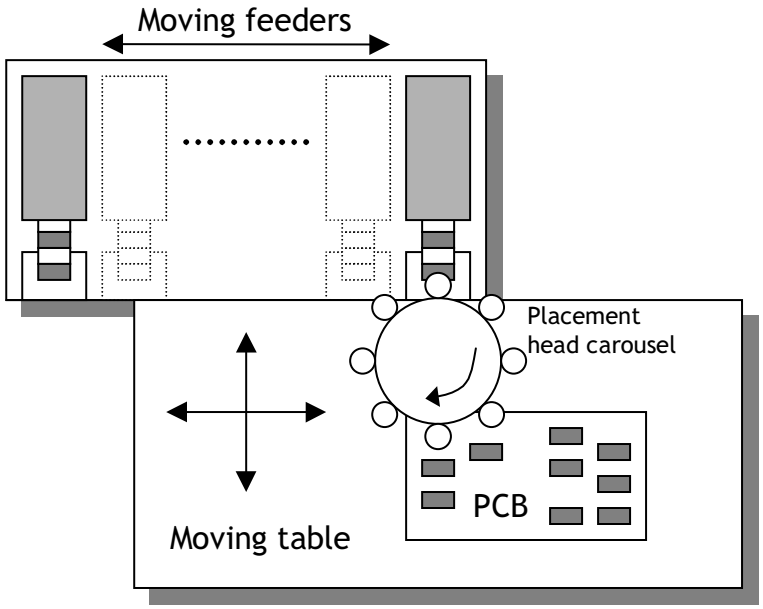


Figure 2: The structure of a rotary turret placement machine [6]

maximum metrics (L_∞) where the largest time component of rotation, x-transfer, y-transfer and feeder transfer gives the final time of a particular placement operation. The control program tries to minimize the printing time and, consequently, arranges the feeder tapes and the pick-up and placement operations so that

1. the feeder and PCB movements are short enough to be carried out during the rotation steps, and
2. if this cannot be realized, the overall length of the movements is short and long jumps occur parallel in both the feeder unit and the holding table.

Obviously, we get accurate time estimates of the placement operations either by performing the operations in the device or by using a high-quality simulator. Because these two estimation techniques are often ruled out for their high cost, we look next at the quality of simple estimates.

3 Linear Estimators for the Processing Time

We have generate a large set of artificial assembly tasks and determined their effective machine control programs with the LEP optimization system. The

optimizer has three operation modes:

Level 1 The optimizer performs a quick-and-dirty control program generation. This option suits for testing prototypes when the amounts of PCBs are small or the result of level 1 optimization are used as an initial solution for higher level optimization. The running time is typically few tens of second and it is limited to one minute.

Level 2 The optimizer is suited for normal production situations when the PCB batches are of a reasonable size. The running times are typically from few minutes and they are restricted to one hour. The optimizer determines four reasonable feeder setups and chooses among them the most promising one.

Level 3 Optimization for mass production which takes typically one work day. The optimizer performs a full-scale optimization of the feeder setup. The difference of the objective function in comparison to the level 2 is of the size one percent. The optimization can be terminated at any time and the system then returns the best solution so far.

Again, we emphasize that when we speak about the "value of the solutions", we refer to the simulator times. When the solutions are carried out in the real world, the observed time differs slightly from the simulator (usually less than two percent) but the difference is stochastic with a zero mean. Nevertheless, the simulator assumes that the machine operates on ideal conditions (e.g., all components are valid and the machine is freshly serviced).

All levels of optimization include the option to use or to avoid the feeder setup optimization. Sometimes some feeder slots should contain fixed component types from the previous production phases and the new components, required by the current PCB, are added to the remaining feeder slots. As a rule, one should perform the feeder allocation optimization whenever possible, since it normally has a significant effect on the production time. On the other hand, the manual resequencing of the component reels on the feeder unit is demanding and (unless it can be performed in parallel to the production of the previous batch) its duration is away from the effective operation time. Thus, there is a tradeoff between the savings in production time and feeder setup time.

The motivation behind this elaboration on optimization of the control code is that the high quality of the control code is the basis for the whole production control and we assume that the placement machines are programmed to work optimally. (N.B. Here 'optimality' refers to *locally* optimal

solutions. The actual global optimum is seldom achieved, since it requires a solution where both the feeder and holding table movements are optimal.)

Because of time restrictions, we use level 1 and level 2 optimized codes for each sample problem (in total 78 problem instances). The optimization is performed in ideal conditions, where the xy-table is moving in full speed (slow-down 0%), the turret cycle time is 100 ms and the table speed and feeder speed are 100 ms. In the first experiment we generate "random" PCBs so that we place n components to a square-like PCB with a side length of ℓ millimeters. To see the effect of the feeder allocation, we draw the components from a set of c different components. The locations of the components on the PCB were chosen by drawing the x and y coordinates independently and uniformly from the interval $[0, \ell]$. In particular, we let $n \in \{50, 100, 400, 700, 1000\}$; $c \in \{1, 30, 100\}$; $\ell \in \{0, 500, 2000, 5000\}$. This gives us in total 58 sample problems of typical sizes.

We would like to point out three properties in our test problems. First, all components are assumed to be full speed parts. This means that the PCBs can be moved at the best rate (with a 0% slow down). It is common that the "slow" components are placed at the end of the placement sequence because after the first slow component, all subsequent component require a lower table speed. This is necessary to avoid displacing the previous slow components on a slippy PCB. Slow components can be accounted by adding new terms to the production time model. However, we omit it because it would make our analysis unnecessary complicated. Second, we assume that all component reels are "narrow" (i.e., a reel takes 2 feeder slots). Third, the usage pattern of different component types are simplified. We assume that the components are selected among a given set of c candidates and each of them has the same probability to be chosen. We return to this assumption later.

Appendix A includes the results for this set of sample data. We observe that the relative difference of level 1 and level 2 solutions is between 0 and 0.12, and the relative difference remains the same for the whole of level times, see Figure 3. Consequently, this motivates to use level 2 optimization whenever possible. We note that the case $c = 1$, $\ell = 0$ corresponds to the case where all components are placed at the same location on the PCB, and no feeder positioning is needed. We observe that in this unnatural deterministic case the production time is linear on the number of points n, see Figure 4.

A more surprising—and, at the first sight, disturbing—observation is that if we fix $n = 100$, $c = 1$ (i.e., there are no feeder movements) and let the board size vary from 0 to 50000, the smallest production time does not occur for the smallest board size 0 but in the size 20000. The main reason for this is the layout of the machine in question. As described in Section 2, the PCBs

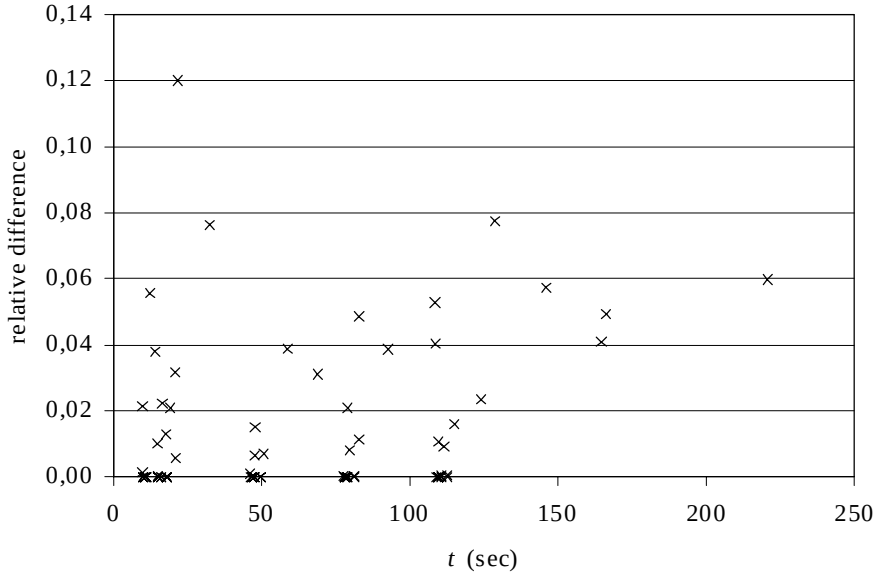


Figure 3: Relative difference $((t_1 - t_2)/t_1)$ for different level 1 values.

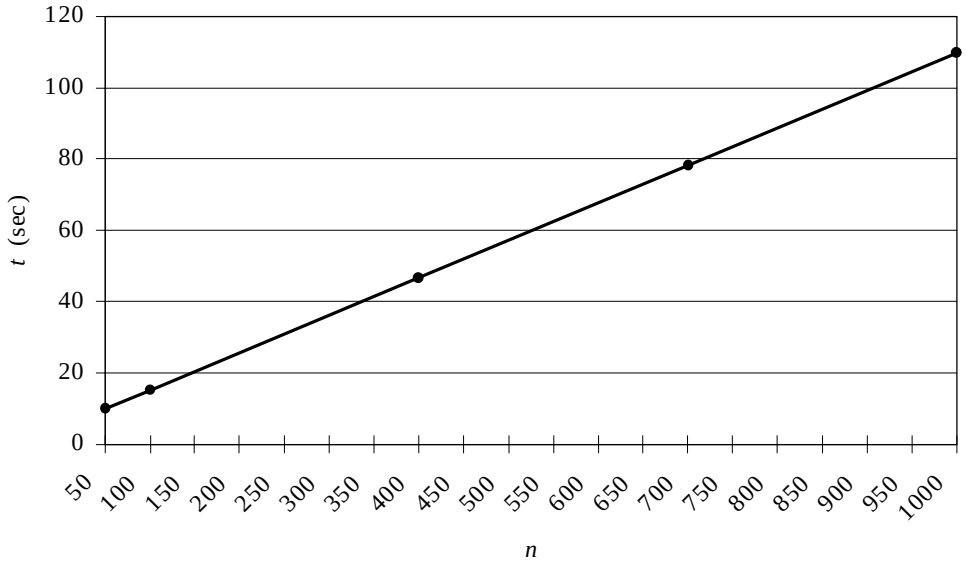


Figure 4: Production time as a function of n when $c = 1$ and $\ell = 0$.

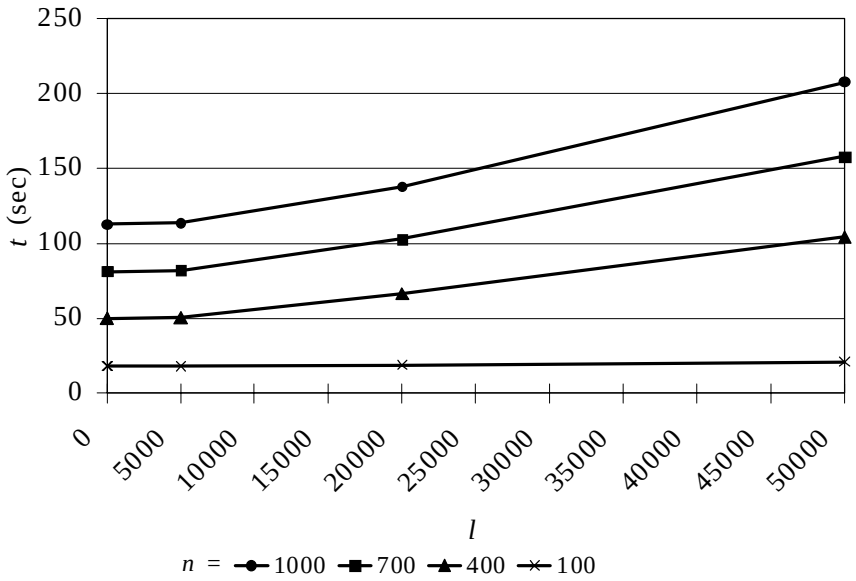


Figure 5: Production time as a function of board size ℓ when $c = 100$ and $n \in \{100, \ldots, 1000\}$.

are transported on a line which is quite far away from the turret. When components are placed on the board, the board is moved properly under the turret. The amount of extra movements depends on the size of the board and the density of the placement locations on the board. As a result, when $\ell = 20000$, these movements are minimal. The effect disappears when the number of different components gets larger, see Figure 5. For a fixed value of c, an increase in the board size causes an increase in printing time, since the xy table movements are longer.

Our sample data show a very clear dependence of the production time on the number of different components (c) when n and ℓ are fixed, see Figure 6. This is due to the increased feeder movements.

We fitted linear regression functions to our data and obtained high coefficient of determination. A simple model

$$t_1 = 0.119 \cdot n + 7.123$$

has a coefficient of determination of 82.4 percent for level 1 production time and

$$t_2 = 0.117 \cdot n + 6.991$$

explains 85.1 percent of the variation for level 2 times.

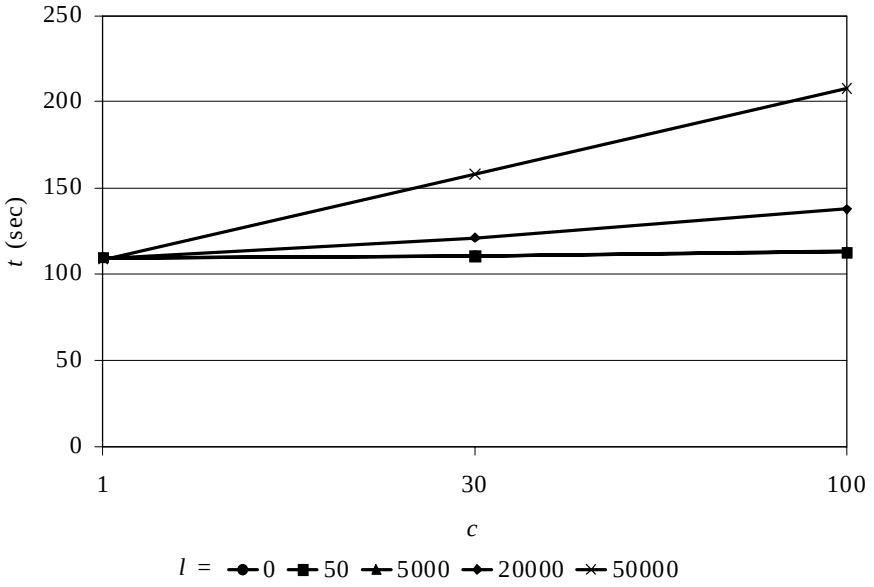


Figure 6: Production time as a function of c when $n = 1000$ and $\ell \in \{0, \ldots, 50000\}$. The times for cases $\ell \in \{0, 50, 5000\}$ are essentially the same.

The linear multiregression models

$$t_1 = 0.117 \cdot n + 0.180 \cdot c + 0.000617 \cdot \ell - 8.097$$

$$t_2 = 0.115 \cdot n + 0.163 \cdot c + 0.000536 \cdot \ell - 6.467$$

have the coefficient of determination 91.0 and 92.2 percent, respectively.

The assumption that each component type is used with the same probability is made to keep the model simple. To see the effect of unequal usage patterns of the components we generated two artificial PCB sets containing 10 layouts each. The first one has the parameters $n = 400$, $c = 30$, $\ell = 20000$ and component occurrences were uniform as above. In the second test set, the component occurrence are nonuniform so that after selection the c components which occur in the board, we place one of them once on the board on a random location. Next, we divide the components into two sets; common components (20 percent of all) and rare components (80 percent of all types). The remaining $n - c$ components are then sampled so that 80 percent of the components are selected among the set of common components and the remaining 20 percent among the rare components. With this restriction we approximate a skewed distribution of component types which is, according to our experience, typical in real-world PCBs.

For the random occurrence model, the average level 1 and level 2 production times are 58.234 and 56.003 (with variance 2.339 and 1.746). Similarly, for the skewed occurrence model these numbers are 55.119 and 52.688 (with variances 0.302 and 0.215). Thus, we conclude that the nonuniform usage patterns give smaller production times than the random model when tested by a (two-tailed heteroscedastic) Student's t-test. Again, this demonstrates clearly that the layout of the PCBs has an impact on the total production time.

4 Conclusions

A fast estimator for the production time is important when solving line balancing problems in multiproduct PCB assembly. We demonstrated that the actual production time depends on several different factors such as the number of components, their types, the boards size, component usage patterns, placement speed, and the rest position of the turret. A linear model using the first three of these operates relatively well. It searches the coefficient of determination 90 percent which still leaves room for error that may risk the results of the overall system. Therefore, this kind of optimization can serve only as an initial step and the final optimization should utilize a simulator for estimating the times. This make the optimization challenging: One has to keep the number of optimization steps low and, at the same time, to strive towards the best possible local minimum as fast as possible. This is possible by using the three different time estimators at different stages of the optimization process.

We must emphasize that in this work we considered one special machine type only. Analysis of several different machine types is a subject for further studies. It is possible that some machine types are simple enough for estimating by analytic functions, and we may find efficient estimators for these machines and reduce the need for the slow simulators.

References

[1] Y. Crama, O. E. Flippo, J. van de Klundert, and F. C. R. Spieksma. The assembly of printed circuit boards: A case with multiple machines and multiple board types. *European Journal of Operational Research*, 98(3):457–72, 1997.

[2] Y. Crama, A. Oerlemans, and F. Spieksma. *Production Planning in Automated Manufacturing*, volume 414 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, 1994.

[3] M. Johnsson. *Operational and Tactical Level Optimization in Printed Circuit Board Assembly*. PhD thesis, University of Turku, 1999. TUCS Dissertations 16.

[4] C. Klomp, J. van de Klundert, F. C. R. Spieksma, and S. Voogt. The feeder rack assignment problem in PCB assembly: A case study. *International Journal of Production Economics*, 64:399–407, 2000.

[5] T. L. Landers, W. D. Brown, E. W. Fant, E. M. Malstrom, and N. M. Schmitt. *Electronics Manufacturing Processes*. Prentice-Hall, Englewood Cliffs, NJ, 1994.

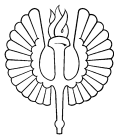
[6] J. Smed, M. Johnsson, T. Johtela, and O. Nevalainen. Techniques and applications of production planning in electronics manufacturing systems. Technical Report 320, Turku Centre for Computer Science, Dec. 1999.

A Results

n	*c*	*l*	*t1*	*t2*	*n*	*c*	*l*	*t1*	*t2*
50	1	0	10,00	10,00	400	100	0	49,76	49,76
50	1	50	10,00	10,00	400	100	50	49,76	49,76
50	1	5000	9,87	9,86	400	100	5000	50,88	50,52
50	1	20000	9,92	9,71	400	100	20000	68,90	66,75
50	1	50000	12,39	11,70	400	100	50000	108,68	104,30
50	30	0	10,89	10,89	700	1	0	78,25	78,25
50	30	50	10,89	10,89	700	1	50	78,25	78,25
50	30	5000	10,88	10,88	700	1	5000	78,10	78,10
50	30	20000	14,13	13,59	700	1	20000	77,65	77,64
50	30	50000	21,76	19,14	700	1	50000	79,09	77,44
100	1	0	15,25	15,25	700	30	0	79,14	79,14
100	1	50	15,25	15,25	700	30	50	79,14	79,14
100	1	5000	15,11	15,10	700	30	5000	79,69	79,04
100	1	20000	14,96	14,81	700	30	20000	92,77	89,18
100	1	50000	17,71	17,49	700	30	50000	128,83	118,84
100	30	0	16,14	16,14	700	100	0	81,26	81,26
100	30	50	16,14	16,14	700	100	50	81,29	81,26
100	30	5000	16,69	16,31	700	100	5000	82,96	82,02
100	30	20000	20,95	20,28	700	100	20000	108,51	102,78
100	30	50000	32,59	30,11	700	100	50000	164,72	157,98
100	100	0	18,26	18,26	1000	1	0	109,75	109,75
100	100	50	18,26	18,26	1000	1	50	109,75	109,75
100	100	5000	18,14	18,14	1000	1	5000	109,60	109,60
100	100	20000	19,30	18,89	1000	1	20000	109,14	109,14
100	100	50000	21,15	21,03	1000	1	50000	109,81	108,63
400	1	0	46,75	46,75	1000	30	0	110,64	110,64
400	1	50	46,75	46,75	1000	30	50	110,68	110,64
400	1	5000	46,60	46,60	1000	30	5000	111,52	110,50
400	1	20000	46,18	46,14	1000	30	20000	124,13	121,22
400	1	50000	48,00	47,28	1000	30	50000	166,29	158,09
400	30	0	47,64	47,64	1000	100	0	112,76	112,76
400	30	50	47,64	47,64	1000	100	50	112,83	112,76
400	30	5000	47,88	47,57	1000	100	5000	115,13	113,29
400	30	20000	58,82	56,54	1000	100	20000	145,97	137,61
400	30	50000	82,99	78,96	1000	100	50000	220,79	207,57

Turku Centre for Computer Science
Lemminkäisenkatu 14
FIN-20520 Turku
Finland

http://www.tucs.fi



University of Turku
- **Department of Mathematical Sciences**

Åbo Akademi University
- **Department of Computer Science**
- **Institute for Advanced Management Systems Research**

Turku School of Economics and Business Administration
- **Institute of Information Systems Science**