Sequencing Robotic Diagnostic Tasks via Optimized Stochastic Policy Trees

András Kovács, Bence Tipary

HUN-REN Institute for Computer Science and Control (SZTAKI)
1111 Budapest, Kende u. 13-17, Hungary
{andras.kovacs,bence.tipary}@sztaki.hu

Abstract

This paper introduces a novel approach to sequencing robotic diagnostic tasks performed on faulty electronic printed circuit board (PCB) products. By assuming that each product has a single fault, at any given moment of the diagnostic process, sequencing must consider only the diagnostic tasks that help identify possible faults not ruled out by the earlier tests. With this, the solution of the stochastic sequencing problem can be encoded into a policy tree that prescribes the next task to execute depending on the results of earlier tests. The paper proposes a local search approach to minimizing the expected duration of the diagnostic process by constructing an initial policy tree using a greedy entropy-per-cost heuristic, and then improving this solution further by a hill climbing search and an adaptation of the insertion neighborhood. An industrial application is presented and first experimental results are reported.

Introduction

Industrial robotics for decades focused on mass production where robots could endlessly repeat their predefined programs, executed with open loop control in perfectly designed environments. This comfortable situation is changing radically with the endeavor to automate complex processes that earlier required human dexterity; and with steps made towards circular economy, where re-manufacturing processes must handle each product individually depending on its condition. Hence, instead of executing off-line programmed robot codes in meticulously designed robot cells, real-time planning techniques based on sensor signals must be adopted on all levels of decision making, including task planning (Johannsmeier and Haddadin 2017), path planning (Bruce and Veloso 2002; Kunz et al. 2010), as well as robot control (Chaumette and Hutchinson 2006; Kim and Croft 2019).

This paper is motivated by an industrial application aimed at automating the diagnosis and repair of high-value PCB products. A novel approach is proposed for sequencing the diagnostic tasks to minimize the expected duration of the diagnostic process. By assuming that each product has a single fault, at any particular moment during the diagnostic process, sequence planning can focus solely on the tasks that help identify possible faults that could not be ruled out based on the results of earlier diagnostic tasks. Ultimately, the diagnostic process can be terminated when the unique fault is unambiguously identified, without executing the remaining tasks. Yet, the actual fault is unknown at the beginning; it is revealed gradually during the process, which gives rise to a stochastic optimization problem.

Approaches to robotic task sequencing typically rely on the well-known travelling salesman problem (TSP) or one of its numerous extensions (Alatartsev, Stellmacher, and Ortmeier 2015; Suarez, Lembono, and Pham 2018; Zahorán and Kovács 2022). Task sequencing specifically for automated in-circuit test probing was investigated recently in (Bonaria et al. 2019). However, all the above deterministic approaches assume that the entire set of tasks must be executed. Various stochastic extensions of TSP were investigated, including probabilistic arc costs (Toriello, Haskell, and Poremba 2014), a random subset of vertices that must be visited (Bertsimas 1988), as well as dynamic variants of vehicle routing problems (Jaillet and Wagner 2008).

From another point of view, the sequencing of diagnostic tasks can be seen as an extension of the binary identification problem (Garey 1972), commonly applied to test sequencing (Raghavan, Shakeri, and Pattipati 1999). Yet, to the best of our knowledge, the current paper is the first to study the combination of the two problems, i.e., test sequencing with sequence-dependent costs, or vice versa, computing the optimal stochastic policy for sequencing robotic diagnostic tasks.

Our contributions include formulating the sequencing of robotic diagnostic tasks as a stochastic optimization problem, encoding the solution into a policy tree, and introducing a local search approach to minimize the expected duration. The algorithm is evaluated in initial computational experiments on random problem instances.

Motivating Industrial Application

In the field of measurement and diagnosis of used PCBs, robotization is getting more and more prominent. However, automation is still very challenging since robotic measurement cells need to be highly flexible to handle a large variety of used, faulty electronic products and diverse failure modes. An experimental robot cell is presented in Fig. 1.

In general, the diagnosis and repair process for an already familiar product type can be described as follows: a PCB product arrives into the shop, accompanied by an error code describing a general failure mode. For the particular fail-

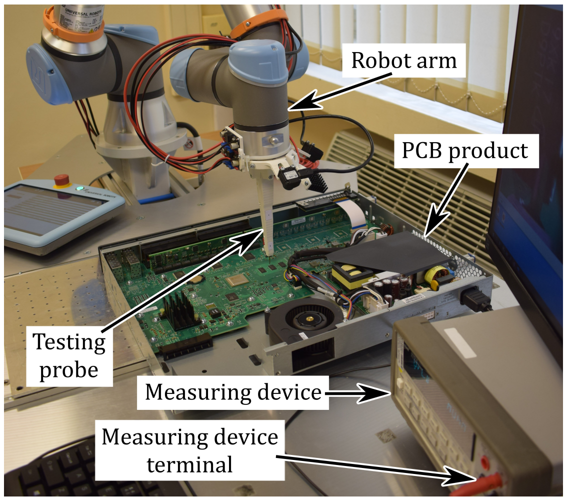

Figure 1: Robotic measurement cell and main components.

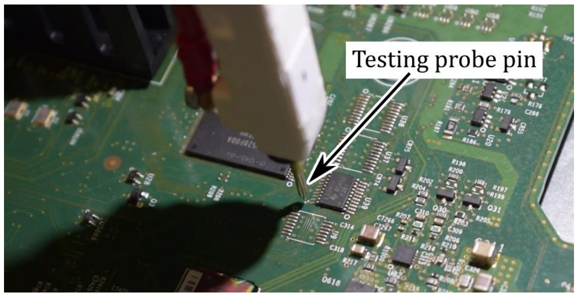

Figure 2: Close-up view on establishing contact between the probe pin and a measurement point on the PCB.

ure mode, there is a prepared diagnostic profile with a finite amount of possible outcomes to uncover possible faults. The diagnostic profile is executed using the robotic measurement system, and based on the outcome, the product is sent to a repair station for component repair or replacement; it is forwarded for further diagnostics; or it gets scrapped.

A diagnostic profile is a collection of specific measurement points on the PCB (via holes, IC legs, etc.) that need to be measured. The electronic characteristics of the faulty PCB are uncovered by connecting the terminal of a measuring device (such as a multimeter or oscilloscope) to these measurement points, and performing the required measurement.

In robotized repair shops, this traditionally manual work can be performed using a robotic measurement process, where the robot arm is equipped with a testing probe (Tipary et al. 2023). The testing probe is connected to the terminal of the measuring device, and it is manipulated by the robot to establish the galvanic contact between the probe pin and the measurement point, as shown in Fig. 2. This robotic process allows experts to focus on diagnostic profile preparation and strategic decisions instead of repetitive diagnostic tasks.

When executing a diagnostic profile, the robot moves the testing probe above the PCB, into the approach position corresponding to the actual measurement point, then feeds forward to establish contact, performs the electronic measurement, retracts to above the PCB, and moves to the next location. During the measurement of a PCB product, software tasks (powering on or off, running diagnosis, etc.) can also be performed between robotic diagnostic tasks.

Decisions on how the products proceed in the shop are made based on fault signatures that are identified during the measurement process. Fault signatures are identified and prepared by experts based on product specification, product-, component- and electronics-related knowledge, as well as repair strategy. As repair shops, and particularly robotized repair shops generally have a large amount of collected data regarding faults and faulty components, they have the knowledge on the different faults, which can be translated to the probabilities of signatures occurring. As repair shops need to digitalize the know-how on the diagnostic process for each managed PCB product and failure mode, experts have a vast amount of data to deal with, and a lot of diagnostic profiles to prepare. Consequently, the automation or facilitation of process planning and sequencing is called for by the industry.

Problem Definition

The diagnostic task sequencing problem aims at finding the most efficient way to identify the single *fault* of a defective product out of the possible K faults, denoted by F_1, F_2, ..., F_K. For this purpose, a robot can perform N *diagnostic tasks*, $T_1, T_2, ..., T_N$, that have binary outcomes: the product either passes the test or not. Each possible fault F_k is unambiguously characterized by a unique, deterministic *signature*, i.e., vector $S_k = (s_{k1}, s_{k2}, ..., s_{kN})$ with $s_{ki} \in \{0, 1\}$ that indicates the binary outcomes of the diagnostic tasks in case of the given fault. The a priori probability of fault k is denoted by p_k, with $\sum_{k=1}^{K} p_k = 1$.

Diagnostic tasks are located at different areas of the product. For the robot, it takes t_{ij} time to execute task j after task i, which includes robot movement from location i to location j, any required instrumentation changeover between the tasks, as well as performing diagnostic task j itself. It is assumed that durations satisfy the triangle inequality. At the beginning and the end of the diagnostic process, the robot departs from, and it returns to its home position, denoted by position T_0. All input data are visualized for a small sample instance in Fig. 3. In the instance, the first fault corresponds to a fully operational product; the second and third to a product with a specific broken component; whereas the fourth to a product that is completely unresponsive due to power supply failure.

The diagnosis of a given fault F_k is accomplished when the robot completes the execution of a subset of the diagnostic tasks $I \subseteq [N]$ such that the outcomes of the performed tasks unambiguously identify the signature of the fault, i.e., $\forall k' \neq k \in [K]\ \exists i \in I : \ s_{k'i} \neq s_{ki}$. Hence, it might not be necessary to execute all tasks to complete the diagnostic process. In a given stage of the diagnostic process, the next task to execute can be selected on the fly, depending on the outcomes of the earlier executed tasks.

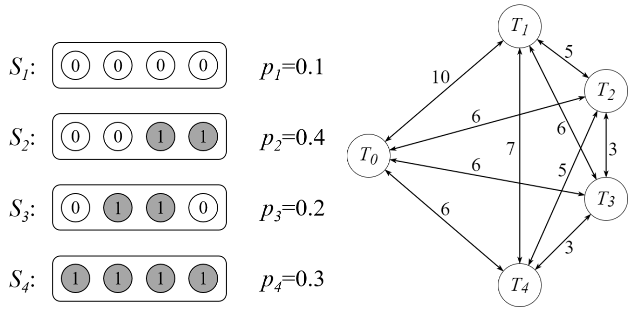

Figure 3: Sample problem instance: signatures and probabilities of faults (left) and distance of diagnostic task locations (right).

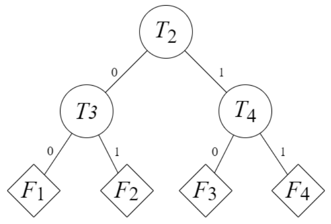

Figure 4: Optimal policy tree for the sample instance.

With the above, the solution of this stochastic optimization problem can be encoded into a binary *policy tree* rooted at the first diagnostic task to be executed. Each non-leaf vertex has two descendants, corresponding to the next task to be executed upon a positive (respectively, negative) outcome of the parent task. The tree has exactly K leaves corresponding to the possible faults. Each vertex v is characterized by its so-called ambiguity set, i.e., the subset of faults possible in that vertex, $F(v)$. If v is the root vertex, then $F(v) = [K]$, whereas $F(v')$ is singleton iff v' is a leaf. Moreover, if v_0 is the parent of v_1 and v_2, then $F(v_0) = F(v_1)\dot{\cup}F(v_2)$, where $\dot{\cup}$ denotes the disjoint union operator.

The total duration of the diagnostic process in case of a specific fault can be computed by summing the task durations along the root-to-leaf path (also taking care of the travel time from the home position to the root task, as well as from the leaf to the home position). The objective is minimizing the expected duration of the diagnostic process. Obviously, the optimal policy tree depends both on the fault probabilities (the most likely faults should be identified as soon as possible) and the distance between task locations (nearby tasks should be executed after each other). The optimal solution of the sample instance is displayed in Fig. 4.

Proposed Solution Approach

Initial Solution

The initial policy tree is constructed using a top-down heuristic that selects the task in the root vertex first, and from any given vertex proceeds to the two children of that vertex. In each step, it selects the diagnostic tasks that extracts the most diagnostic information at in shortest possible time. Formally, in vertex v, it selects the task i that maximizes the

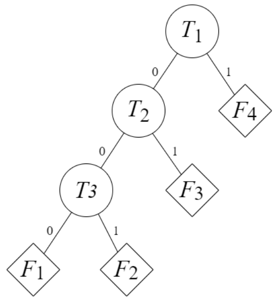

Figure 5: Application of the insertion-with-duplication operator: task T_1 is inserted above the root and redundant vertices are removed.

entropy-per-cost ratio, calculated as

$$\text{EPC}(v,i) \;=\; \frac{-p^0_{vi}\log_2 p^0_{vi} \;-\; p^1_{vi}\log_2 p^1_{vi}}{t_{ji}}$$

where the $p^{\alpha}_{vi} = \sum_{k\in F(v):\; s_{ki}=\alpha} p_k / \sum_{k\in F(v)} p_k$ is the probability of passing ($\alpha = 0$) or failing ($\alpha = 1$) diagnostic task i in vertex v, and hence, the numerator is the Shannon entropy of executing the diagnostic task i in vertex v. The denominator is the time required for extracting that information, where j is the task executed in the parent vertex. This step is iterated on each branch of the policy tree until the ambiguity set $F(v)$ is singleton, which means that the given fault mode is unambiguously identified, and the current vertex is a leaf of the policy tree.

Local Search

The initial solution is improved by a hill climbing search over an *insertion neighborhood* adapted to the stochastic problem and the policy tree representation of the solution. Namely, the algorithm tries to insert each task T_i above each vertex v into the policy tree. As a side effect of insertion, some tasks below the inserted vertex may become redundant, i.e., their entropy may decrease to zero. Two versions of the insertion operator are defined for managing this phenomenon.

The *insertion-with-duplication* operator creates two copies of the sub-tree below vertex v at the place of both children of the inserted vertex, and deletes potential redundant vertices from the sub-trees. This is illustrated in Fig. 5, where task T_1 is inserted above the root of the tree displayed in Fig. 4. Two copies of the original tree are created below the inserted vertex. In the left sub-tree, task T_2 is necessary to differentiate faults $\{F_1, F_2\}$ from F_3, T_3 is required to tell F_1 apart from F_2, but task T_4 can be omitted. In the right sub-tree of the inserted vertex, the only remaining fault is F_4, and hence, all further diagnostic tasks can be dropped.

In contrast, the *insertion-with-regeneration* operator simply uses the entropy-per-cost heuristic to generate the subtrees below the inserted vertex from scratch.

In each iteration, the neighbor that yields the highest improvement is selected as the next incumbent solution, and this step is iterated until a local optimum is reached where no neighbor improves on the incumbent solution.

Table 1: Experimental results.

Prob	MILP			Initial		Local search	
	Sol	Gap	Time (s)	Sol	Gap	Sol	Gap
diag5x5_1	12.97	0.00%	5.7	13.31	2.66%	12.97	0.00%
diag5x5_2	13.23	0.00%	9.5	16.00	20.93%	13.23	0.00%
diag5x5_3	10.29	0.00%	3.0	12.41	20.57%	10.29	0.00%
diag5x5_4	6.42	0.00%	2.9	6.42	0.00%	6.42	0.00%
diag5x5_5	8.00	0.00%	3.0	8.00	0.00%	8.00	0.00%
diag10x10_1	7.49	14.09%	600.0	6.94	5.68%	6.57	0.00%
diag10x10_2	11.77	32.20%	600.0	9.74	9.32%	8.91	0.00%
diag10x10_3	11.07	42.69%	600.0	8.62	11.17%	7.76	0.00%
diag10x10_4	13.10	51.49%	600.0	9.29	7.46%	8.65	0.00%
diag10x10_5	12.42	64.10%	600.0	7.92	4.63%	7.57	0.00%

Experimental Results

Although the presented robotic diagnostic cell is deployed at the industrial partner, it currently employs a fixed task sequence specified by human experts, executing all tasks in the sequence independently of the measurement results. The task sequencing approach proposed in this paper, implemented in C++, was evaluated in simulation experiments on randomly generated problem instances. In the experiments, it was compared to an exact solver based on a deterministic equivalent mixed-integer linear program (MILP) model in FICO Xpress, omitted here due to space restrictions.

The results are presented in Table 1, where each row stands for an individual problem instance. In the first column, instance names also indicate problem size: $N = K = 5$ for small instances and $N = K = 10$ for large instances. Subsequent columns display solution values, i.e., expected durations (Sol); gaps compared to the best solution delivered by any of the algorithms (Gap); and computation times (Time) for the MILP-based exact solver, for the initial solution constructed using the entropy-per-cost heuristic, and for the complete local search approach. Computation times are omitted for local search, since it terminated in a local minimum for every instance very quickly, in less than 35 milliseconds.

For all small instances, Xpress found optimal solutions in 2.9-9.5 seconds. In contrast, for all medium size instances, it hit the time limit of 600 s and constructed feasible, but poor quality solutions. Also, it could not provide any reasonable lower bounds. Accordingly, the optimal solution for the large instances remains unknown.

The performance of the initial solution heuristic varied significantly over the different instances, and the gap exceeded 20% for two small instances. Local search improved those initial solutions significantly. For all small instances, it found the proven optimal solution, whereas for the medium size instances, it constructed the best known solution, outperforming both the MILP solver and the constructive heuristic.

In the real industrial application, we expect instances with 15-25 tasks and 20-40 possible faults. The proposed approach was tested on random problems of that size, and it was found to terminate with local optima in at most one second. At the same time, we do not have any means yet to verify the quality of those solutions. Currently, we are in the process of data acquisition and fitting industrial data into our model.

Conclusions and Future Research

This paper defined the robotic diagnostic task sequencing problem, and proposed a simple but efficient local search method for computing a solution in the form of a stochastic policy tree. Compared to classical deterministic task sequencing approaches that compute a fixed sequence of tasks, the proposed technique brings huge savings in expected duration in applications where the single fault of the product can be identified by executing a small subset of the possible diagnostic tasks.

In initial computational experiments, the proposed approach was compared to an exact solver on small and medium-size, randomly generated problem instances. Local search found the exact optimal solution for all instances where that optimal solution is known, whereas it outperformed the exact solver on the more challenging instances. A future research step of utmost importance will be the evaluation of the approach on instances stemming from the actual industrial application.

Intriguing directions for future research include relaxing the singe fault assumption, extending the model to diagnostic tasks with stochastic outcomes, as well as to fault signatures involving logical conditions. Another natural extension may involve diagnostic tasks executable using different robot poses. The latter implies that the sequencing problem is combined with the selection of robot poses, similarly to the selection of a vertex from each class in GTSP.

Acknowledgements

This research has been supported by the TKP2021-NKTA-01 NRDIO grant on "Research on cooperative production and logistics systems to support a competitive and sustainable economy" and by the European Union within the framework of the National Laboratory for Autonomous Systems (RRF-2.3.1-21-2022-00002).

$$p_{vi}^{\alpha} = \frac{\sum_{k \in F(v):\, s_{ki}=\alpha} p_k}{\sum_{k \in F(v)} p_k}$$

References

Alatartsev, S.; Stellmacher, S.; and Ortmeier, F. 2015. Robotic task sequencing problem: A survey. *Journal of Intelligent & Robotic Systems*, 80(2): 279–298.

Bertsimas, D. 1988. *Probabilistic combinatorial optimization problems*. Ph.D. thesis, Operations Research Center, MIT, Cambridge.

Bonaria, L.; Raganato, M.; Reorda, M. S.; and Squillero, G. 2019. A dynamic greedy test scheduler for optimizing probe motion in in-circuit testers. In *2019 IEEE European Test Symposium (ETS)*, 1–2.

Bruce, J.; and Veloso, M. 2002. Real-time randomized path planning for robot navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, 2383–2388.

Chaumette, F.; and Hutchinson, S. 2006. Visual servo control. I. Basic approaches. *IEEE Robotics & Automation Magazine*, 13(4): 82–90.

Garey, M. R. 1972. Optimal binary identification procedures. *SIAM Journal on Applied Mathematics*, 23(2): 173–186.

Jaillet, P.; and Wagner, M. R. 2008. Online vehicle routing problems: A survey. In Golden, B.; Raghavan, S.; and Wasil, E., eds., *The vehicle routing problem: Latest advances and new challenges*, 221–237. Springer US.

Johannsmeier, L.; and Haddadin, S. 2017. A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes. *IEEE Robotics and Automation Letters*, 2(1): 41–48.

Kim, J.; and Croft, E. A. 2019. Online near time-optimal trajectory planning for industrial robots. *Robotics and Computer-Integrated Manufacturing*, 58: 158–171.

Kunz, T.; Reiser, U.; Stilman, M.; and Verl, A. 2010. Real-time path planning for a robot arm in changing environments. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5906–5911.

Raghavan, V.; Shakeri, M.; and Pattipati, K. 1999. Test sequencing problems arising in test planning and design for testability. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 29(2): 153–163.

Suarez, F.; Lembono, T.; and Pham, Q. C. 2018. RoboTSP – A fast solution to the robotic task sequencing problem. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 1611–1616.

Tipary, B.; Juniki, Á.; Erdős, F. G.; Takács, E.; and Németh, K. 2023. Development of a novel visual servoing probe test method for fault diagnosis of printed circuit boards. *IFAC-PapersOnLine*, 56(2): 5248–5254. 22nd IFAC World Congress.

Toriello, A.; Haskell, W. B.; and Poremba, M. 2014. A dynamic traveling salesman problem with stochastic arc costs. *Operations Research*, 62(5): 1107–1125.

Zahorán, L.; and Kovács, A. 2022. ProSeqqo: A generic solver for process planning and sequencing in industrial robotics. *Robotics and Computer-Integrated Manufacturing*, 78: 102387.