Proceedings of the Ninth International
Symposium on Combinatorial Search (SoCS 2016)

Batch Repair with Heuristic Search

Hilla Shinitzky and **Roni Stern** and **Meir Kalech**
Ben Gurion University of the Negev
Beer Sheva, Israel

Troubleshooting is the task of fixing an abnormally behaving system. Many troubleshooting algorithms employ a diagnosis engine that returns a set of *diagnoses* Ω and their likelihoods ($p(\cdot)$), where a diagnosis $\omega \in \Omega$ is a set of components that may have been faulty and its likelihood $p(\omega)$ is the probability that ω is all the faulty components in the system. The Batch Repair Problem (BRP) is a recently introduced troubleshooting problem in which Ω and $p(\cdot)$ are given as input and the task is to minimize the *total repair costs* incurred until the system is fixed (Stern, Kalech, and Shinitzky 2016). Unlike traditional troubleshooting problems, the total repair costs considered in BRP includes (1) the cost of repairing each faulty component (referred to as the component repair cost) and (2) additional repair overhead costs (denoted $cost_{repair}$). When the repair overhead is more expensive than the component repair cost then it may be more efficient to repair a *batch of components in a single repair action*.

BRP can be modeled as a Markov Decision Process (MDP), where the actions are the possible repair actions and the uncertainty is due to not knowing which components are faulty (and thus, repairing a set of component may or may not fix the system). However, the number of actions in each state is exponential in the number of components, as any batch of components may be considered.

As an alternative, prior work modeled BRP as a combinatorial optimization problem, searching in the combinatorial space of possible repair actions for the *best* repair action to perform next (Stern, Kalech, and Shinitzky 2016), where *best* is measured according to a utility function that estimates current and future wasted repair costs. There are two challenges in implementing this approach: (1) how to define such a utility function, and (2) how to efficiently search for the repair action that maximizes this function.

Utility Functions

Stern et al. (2016) proposed the *wasted cost* BRP utility functions. These functions consist (1) **false positive costs** ($cost_{FP}$), which are the costs incurred by repairing components that are not really faulty, and (2) **False negative costs** ($cost_{FN}$), which are the overhead costs incurred by future repair actions. The following general formula computes the expected wasted costs of a repair action that repairs the set of components γ

$$C_{WC} = cost_{FP}(\gamma) + (1 - \texttt{SystemRepair}(\gamma)) \cdot cost_{FN}(\gamma)$$

where $\texttt{SystemRepair}(\gamma)$ is the probability that Repair(γ) will fix the system (`SystemRepair( )` can be derived automatically from Ω and $p(\cdot)$ (Stern et al. 2015)). The values of $cost_{FP}$ and $cost_{FN}$, however, are not known, and thus one cannot compute C_{WC}. Wasted cost utility functions estimate C_{WC} by estimating $cost_{FP}$ and $cost_{FN}$. It is reasonable to estimate the false positive costs by $\widehat{cost}_{FP}(\gamma) = \sum_{C \in \gamma}(1 - F(C)) \cdot cost(C)$ where $F(C)$ is the likelihood that component C is faulty (computed from Ω and $p(\cdot)$).

Estimating $cost_{FN}$ requires considering the future actions of the repair algorithm. Two approaches were suggested: (1) *Optimistic* (Opt.), denoted $cost^o_{FN}$, considers the best case, where only one additional repair action would be needed, incurring a single additional overhead cost, and (2) *Pessimistic* (Pess.), denoted $cost^p_{FN}$, embodies the other extreme, in which we assume that every faulty components will be repaired in the future in a single repair action, incurring one $cost_{repair}$ per faulty component. Since we do not know the number of faulty components, we use the expected number of faulty components according to the health state, resulting in $cost^p_{FN}(\gamma) = cost_{repair} \cdot \sum_{c \notin \gamma} F(c)$.

Enhanced Wasted Cost Utility Function. Both optimistic and pessimistic functions do not explicitly consider future false positive costs, i.e., the cost that will be wasted in the future when repairing component that should not have been repaired. To account for the future false positive costs, we added the same computation computed for the current false positive costs, but over all the component not chosen by the current repair action. So, for a batch repair action γ, we estimated the future false positive costs, denoted $cost_{FFP}(\gamma)$, as follows:

$$cost_{FFP}(\gamma) = \sum_{C \notin \gamma}(1 - F(C)) \cdot cost(C)$$

The resulting wasted cost utility function, which we refer to as "FFP enhanced", is given by:

$$C_{WC} = cost_{FP}(\gamma) + (1 - \texttt{SystemRepair}(\gamma)) \cdot (cost^p_{FN}(\gamma) + cost_{FFP}(\gamma) \quad (1)$$

Search Algorithms

For a given utility function, we still need to solve the underlying search problem of finding the repair action that maximizes $u(\cdot)$. We call this the ***BRP***$_S$ problem.

The search space for the ***BRP***$_S$ problem, is a lattice of all possible subsets of the components not repaired so far. Even for systems with only hundreds of components this search space becomes extremely very large. Since Ω contains all diagnoses we limit the ***BRP***$_S$ search space to only consider unions of diagnoses. (this was found empirically to improve the search in most cases). We considered three types of search algorithms for solving the ***BRP***$_S$ problem: (1) a depth-limited breadth-first search (depth-limited to avoid memory and time explosion), (2) a hill climbing (HC) variant in which a batch repair action iteratively, starting from an empty set of components and adding in every iteration the single component that reduces the most the used utility function until there is no component left that reduces the utility function, and (3) an A* (Hart, Nilsson, and Raphael 1968) variant that finds optimal solutions.[1] Implementing A* for ***BRP***$_S$ is challenging since A* requires an admissible heuristic function and because the ***BRP***$_S$ search space it is not monotone, as the cost of nodes along a path in the search space may decrease (Stern et al. 2014).

To meet this challenges, we define A* slightly differently for ***BRP***$_S$. Instead of the path-based $g(n)$ and $h(n)$, we use the following state-based values $cost(n)$ and $b_L(n)$, where $cost(n)$ is the cost of n (according to the utility function used), and $b_L(n)$ is a heuristic function that estimates the cost of the lowest cost node in the subtree of the ***BRP***$_S$ search space rooted at n (including n). We say that $b_L(n)$ is admissible if is a lower bound on the value it estimates, i.e., $b_L(n) \leq \min_{n'}\{cost(n')|n \subseteq n'\}$.

Running A* with $cost(n)$ and an admissible $b_L(n)$ is done as follows. In every iteration the node with the minimal $b_L(n)$ is expanded. If $cost(n)$ is smaller than the incumbent solution then the incumbent solution is updated. The search continues until the cost of the incumbent solution is lower than or equal to the minimal $b_L(n)$ that is expanded.

We construct an admissible heuristic function for ***BRP***$_S$ by first proposing a function $b_L(n, i)$ that lower bounds the cost of all sets of components that contain the components in n and i additional components. We call this an *i-level admissible* heuristic function. Then an admissible $b_L(n)$ takes the minimum over the values of the i-level admissible heuristic functions for any i. Constructing i-level admissible functions can be done by lower bounding the false positive costs and the false negative costs that can be added by i components independently, and then summing these bounds up.

Experimental Results

We evaluated the different utility functions and search algorithms on a diagnosis benchmark that consists Boolean circuits systems (Hansen, Yalcin, and Hayes 1999; Brglez, Bryan, and Kozminski 1989) and a random selection from Feldman et al.'s (2010) observation set. The prior probability of each gate to be faulty was set to 0.01 and a single diagnosis for each observation served as the injected faults. Component repair cost was set to 5, and we experimented with repair overhead costs of 10 and 25. A timeout of 5 minutes was set in every iteration.

		Overhead = 10				Overhead = 25			
	Alg.	A*	HC	B(1)	B(2)	A*	HC	B(1)	B(2)
c432	Opt.	79	65	50	51	133	102	94	84
	Pess.	80	63	51	51	128	95	84	65
	FFP	57	55	52	**42**	85	**72**	99	68
74182	Opt.	56	51	56	54	72	76	97	82
	Pess.	57	51	58	56	75	68	97	73
	FFP	**48**	54	58	**48**	**67**	69	97	74
c880	Opt.	93	80	117	114	163	128	220	207
	Pess.	95	76	119	118	168	118	226	207
	FFP	**73**	74	111	90	119	**94**	205	149

Table 1: Average cost incurred until the system is fixed

Table 1 shows the average cost incurred over all repair actions until the system is fixed. The rows shows results for the different utility functions and benchmark systems (c432, 74182, and C88). The columns are for the different search algorithms, where A*, HC, B(1), and B(2), correspond to our variant of A*, hill climbing, and breadth-first search with one or two depth bound, respectively. Results are given for overhead cost 10 and for 25, and the best result in every configuration is bolded.

A clear trend that can be observed in the results is that the FFP enhanced wasted cost function is in general better than both Opt. and Pess. This result is relatively consistent over all search algorithms. Regarding search algorithms, we observe that A* is superior on the largest system (c880) while BrFS worked surprisingly well for the smaller systems (74182 and c432). Thus, in general, we did not observe any universal winner, suggesting both room for future research as well as demonstrate the potential of a heuristic search approach for this problem.

References

Brglez, F.; Bryan, D.; and Kozminski, K. 1989. Combinatorial profiles of sequential benchmark circuits. In *IEEE International Symposium on Circuits and Systems*, 1929–1934.

Feldman, A.; Provan, G.; and van Gemund, A. 2010. Approximate model-based diagnosis using greedy stochastic search. *Journal of Artificial Intelligence Research (JAIR)* 38:371.

Hansen, M. C.; Yalcin, H.; and Hayes, J. P. 1999. Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering. *IEEE Des. Test* 16:72–80.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on* 4(2):100–107.

Stern, R. T.; Kiesel, S.; Puzis, R.; Felner, A.; and Ruml, W. 2014. Max is more than min: Solving maximization problems with heuristic search. In *Symposium on Combinatorial Search (SoCS)*.

Stern, R. T.; Kalech, M.; Rogov, S.; and Feldman, A. 2015. How many diagnoses do we need? In *AAAI*, 1618–1624.

Stern, R.; Kalech, M.; and Shinitzky, H. 2016. Implementing troubleshooting with batch repair. In *AAAI*.

[1]Note that an optimal solution to ***BRP***$_S$ is not necessarily an optimal solution to the BRP problem.