Converging on the Optimal Attainment of Requirements

Martin S. Feather
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr
Pasadena CA 91109-8099
Martin.S.Feather@Jpl.Nasa.Gov

Tim Menzies
Lane Department of Computer Science &
Electrical Engineering,
West Virginia University
PO Box 6109, Morgantown, WV 26506-6109
tim@menzies.com

Abstract

Planning for the optimal attainment of requirements is an important early lifecycle activity. However, such planning is difficult when dealing with competing requirements, limited resources, and the incompleteness of information available at requirements time.

A novel approach to requirements optimization is described. A requirements interaction model *is executed to randomly sample the space of options. This produces a large amount of data, which is then condensed by a* summarization tool. *The result is a small list of critical decisions (i.e., those most influential in leading towards the desired optimum). This focuses human experts' attention on a relatively few decisions and makes them aware of major alternatives.*

This approach is iterative. Each iteration allows experts to select from among the major alternatives. In successive iterations the execution and summarization modules are run again, but each time further constrained by the decisions made in previous iteration. In the case study shown here, out of 99 yes/no decisions (approximately 10^{30} possibilities), five iterations were sufficient to find and make the 30 key ones.

1. Introduction

Projects that seek to develop complex systems are almost always constrained by limited resources. Resources include development resources (e.g., schedule, budgets, availability of personnel) and product resources (e.g., memory, bandwidth, power). These constraints usually mean that only a subset of all the desired requirements can be attained. Competitive pressures drive projects to seek optimality goals within this constrained space – i.e. to find the *most* requirements for a given set of resources, the *least* resources to attain a given set of requirements, or some combination of these goals.

The importance of this issue has been recognized and studied by the requirements community in recent years. For example:

- Karlsson & Ryan developed a "cost-value" approach to prioritizing requirements [7]. At the heart of their approach is a cost-value diagram, which plots each requirement's relative value and implementation cost, facilitating the selection of an appropriate subset of requirements. They employ the Analytic Hierarchy Process to arrive at the relative value and cost figures for each requirement
- The WinWin project [1] supports multiple stakeholders to identify conflicts between their respective evaluations of requirements, and to locate feasible solutions that are mutually satisfactory combinations of requirements. A recent experience report [6] indicates that the automated aids they have built to support this approach are successful at identifying more issues and options than would be possible by a manual treatment alone.
- Influence diagrams (a form of Bayesian nets) are used in [2] to compute the utility of requirements that are candidates for inclusion in the next release of a piece of software. This enables decision makers to take into account a wide variety of factors contributing to the feasibility of including each requirement.

The approaches cited above essentially assign or calculate cost and benefit figures for individual requirements. The situation becomes more complex if there are significant interactions among requirements, for example, if two requirements can be achieved by sharing the same solutions to sub-problems then the cost of attaining both of them may be significantly less than the sum of their individual costs. Representing and reasoning about requirements *interaction* is another emergent theme within the requirements engineering community. Robinson et al's survey [16] terms this "Requirements Interaction Management", while van Lamsweerde's mini-tutorial [18] refers to "Goal-Oriented Requirements Engineering". Work in this area is founded upon building models of how requirements interrelate – how they decompose, how they are implemented, how they support or contradict one another, etc.

Combining these two themes suggests the following approach:

> *Navigate through alternatives in the cost-benefit tradeoff space, where the cost-benefit figures for the various alternatives are derived from requirements interaction models.*

This paper describes tools to support such navigation. The distinguishing features of our approach are the *type* and *size* of the models we explore. The methods cited above apply to specific kinds of models and require specialized tool support. Also, they typically are applied to small models[1]. In contrast, this paper explores an approach suited to the exploration of *large* models of *any type*. For example, in the case study shown below, some 99 risk mitigation actions were being debated; i.e. $2^{99} \approx 10^{30}$ possible sets of decisions. Further, our exploration tools impose minimal restrictions on the type of model being explored. We just assume that there exists an executable *requirements interaction model* and a *summarization tool*. The former can be of any form, as long as it can be used to generate cost and benefit figures from some model of how requirements interact. The latter is used to extract *summary conclusions* from multiple examples of decisions, each with its cost and benefit figures. These summary conclusions guide the human experts to focus on the relatively few critical decision alternatives, and select accordingly. As they do so, they can bring to bear additional knowledge that they might not have included within the requirements interaction model (e.g., knowledge of incompatibilities between certain decisions) to help them make their selection.

Our approach is to follow the iterative cycle of execution, summarization and decision shown in Figure 1. The requirements interaction model, built by humans, is used to *grow* the space of options, the computer *culls* the less useful of these, and the experts make the final decisions. This can be a fruitful partnership. Focusing experts attention on the relatively small number of most

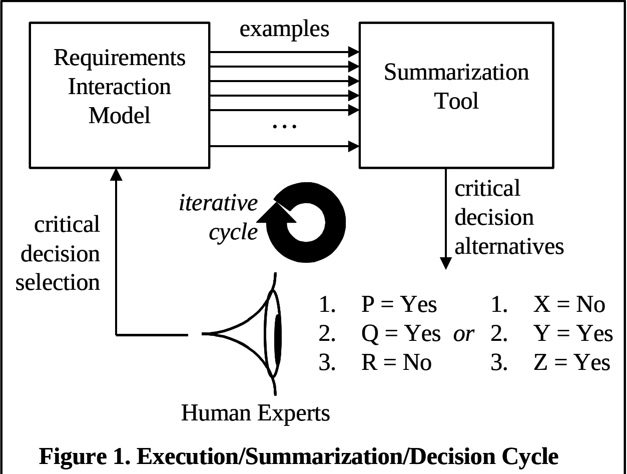
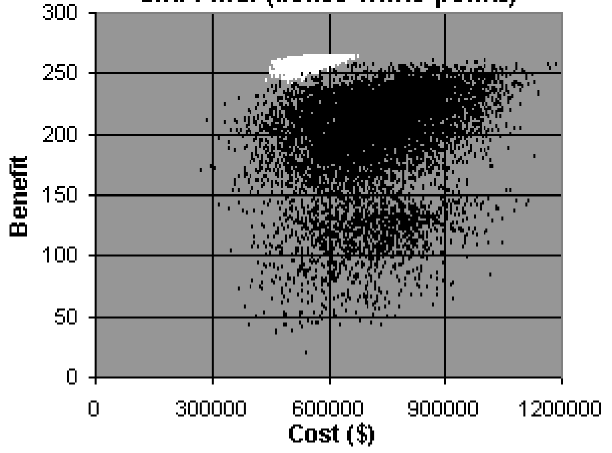


Figure 1. Execution/Summarization/Decision Cycle

critical decision alternatives makes much more effective use of their skill and knowledge. Repeating this cycle allows the iterative approach to the end goal of a decision that is optimal (or near optimal) within the space of feasible decisions.

We demonstrate the feasibility of this approach instantiated on a non-trivial requirements interaction model developed and used at NASA. Sections 2 and 3 describe the requirements interaction model and the summarization tool (respectively) used in our case study. Section 4 shows how the interaction model can be linked to the summarization tool. Section 5 presents our large-scale pilot study that studied a space of 99 possible risk mitigations. In five iterations of the execution / summarization / decision cycle we were able to converge to a desirable region within the cost/benefit space. This is illustrated in Figure 2 where "Benefit" is a measure of requirements attainment and "Cost" is sum of the costs of applying the selected mitigations. Executing the interaction model on randomly selected mitigations generates a sampling of the entire space of solutions (the widely dispersed black points in Figure 2). Executing the interaction model on the mitigation selection recommended by our iterative process generates a sampling of the optimized space of solutions (the dense region of white points in Figure 2). The external validity and applicability of this result is discussed in Section 6.

Figure 2. Initial (scattered black points) and Final (dense white points)



2. DDP: A requirements interaction model

This section introduces the requirements interaction model. We describe the modeling framework, and then the characteristics of a real-world model built within this framework that we used for our large-scale pilot study.

Our requirements interaction modeling framework is the NASA-developed Defect Detection and Prevention (DDP) process and tool for risk assessment, planning and management [4]. DDP deals with requirements, risks and risk mitigations. Risks are quantitatively related to

[1] For example, the reported case studies of the soft goal approach of Mylopoulos and Chung [14, 3] explore a small number of major alternatives; e.g., between one of four major architectural styles for the NFR-Assistant Case tool itself [17].

requirements, to indicate how much each risk, should it occur, impacts each requirement. Mitigations are quantitatively related to risks, to indicate how effectively each mitigation, should it be applied, reduces each risk. A set of mitigations achieves benefits (requirements are met because the risks that impact them are reduced by the selected mitigations), but incurs costs (the sum total cost of performing those mitigations). The main purpose of DDP is to facilitate the judicious selection of a set of mitigations, attaining requirements in a cost-effective manner. DDP has the capability to represent and reason simultaneously with a multitude of mitigations, and their effectiveness at reducing multiple risks. In actual usage, DDP application sessions have dealt with up to 150 each of requirements, risks and mitigations.

Manual exploration of the space of possible mitigation alternatives is a daunting challenge. Complications stem from the interactions within the DDP model – a given risk may impact multiple requirements, and a given requirement may be impacted by multiple risks. Likewise, a given risk may be addressed by multiple mitigations, and a given mitigation may address multiple risks. A modest cost mitigation that addresses a serious risk might seem promising, but if that risk will be adequately addressed by other mitigations that will need to be chosen anyway (primarily to address other risks), then the seemingly promising mitigation may be superfluous. The scale of the models with which we deal compounds the difficulty of manual exploration. For example, if there were 100 mitigations to choose from, the number of possible selections would be 2^{100} (approximately 10^{30}). Automation capable of finding optimal, or near-optimal, solutions is needed. In actual DDP applications (to hardware, software, and hardware-software combinations) the lack of any such automated capability has forced human users to manually pick their suite of risk mitigations. Various graphical presentations of information, automatically computed cost and risk figures, etc., guide them as they do so [5], but it has remained a primarily manual activity. This has several obvious drawbacks – manual selection takes time, and may well fail to find anything near an optimal solution.

For our pilot study, we used a real-world model developed in the DDP framework. We describe its salient characteristics next.

Domain: the study that gave rise to this data was an evaluation of a promising piece of research-quality spacecraft technology. The purpose of the evaluation was to identify the risks that would arise in maturing this technology to flight readiness, and what mitigations could be identified to address those risks in a cost-effective manner. The proprietary nature of the technology precludes discussion of the specifics, which are in any case irrelevant to the focus of this paper.

Scale: NASA experts used DDP to build a network connecting 32 requirements, 69 risks and 99 mitigations. The network contains numerous interaction details: 352 times, the experts commented on how mitigations could reduce risks; 440 times on how risk could damage requirements.

Raw "Benefit" Data: the data that populated the DDP model was the combination of inputs from multiple experts – mission scientists who understood the science requirements driving the need for the technology (e.g., performance metrics), spacecraft engineers who understood the context in which the technology would have to function (e.g., temperature, radiation), subsystem engineers who understood the challenges of matching the novel technology with the various other spacecraft components (e.g., power supply).

Raw "Cost" Data: At the time of the model construction, the group of experts did not assign cost figures to the individual mitigations. They performed their study using DDP to compute the "benefit" side of the equation (i.e., selected mitigations that would reduce risks and thereby lead to attainment of the requirements), and mentally kept track of the cost implications of mitigation selections. In part this was due to the lack of capabilities in DDP at that time to make use of costs. An expert knowledgeable of the expense of the various mitigations added cost figures later. It is this augmented dataset that we used for the collaborative study.

3. TAR2 treatment learning: a summarization tool

Classical machine learning (e.g. C4.5 [15]) can be applied to learn implications between attribute ranges and results (e.g.):

`X>1 and Y<0 implies class=highCostProject`

However, if applied to a non-trivial requirements interaction model a large number of such implications result. Some form of summarization is required. One way to do this is to study pairs of rules that lead to different results and reporting the changes to attribute ranges that change (e.g.) a `highCostProject` into a *`lowCostProject`*. TARZAN implemented such a search as a post-processor to C4.5 [10]. TAR2 performs the same search directly, without needing C4.5 [11]. Starting with examples, TAR2 finds range settings that are highly associated with some "good" outcome (e.g *`lowCostProject`*.) and not highly associated with some "bad" outcome (e.g. `highCostProject`). TAR2 outputs implications of the form (e.g.)

`X>1 and Y<0 implies less "bad" and more "good"`

where "less" and "more" are measures of the change in the frequency of "good" and "bad" before and after applying `X>1 and Y<0` to the examples. The set of attribute ranges (X>1 and Y<0) is called a "treatment". Such treatments are the constraints that TAR2 is proposing on future actions in order to increase the chances of less "bad" and more "good".

TAR2 runs in two passes. In the first pass, a set of promising attribute ranges is discovered. To find such a "promising range", the association and negative association with "good"/ "bad" behavior (respectively) is computed for all ranges. A promising range scores outstandingly high on this scale (for more details, see [11]). In the second pass, all subsets of these promising attribute ranges are tested. Clearly, the second pass is exponential on the number of outstandingly promising ranges. In practice (e.g. the case study discussed below), the number of such outstandingly promising ranges is small enough to be tractable.

This method has demonstrated its utility in a number of widely differing domains, including the COCOMO risk model [10], and CMM level 2 [12]. In those case studies, the summarizations offered by TAR2 were surprisingly succinct. TAR2 could explore megabytes of data to return a single rule describing the *least* action that *most* changes the frequency of "good" and "bad" outcomes.

4. Combining the requirements interaction model with the summarization tool

A requirements interaction model constructed within DDP takes as input a set of decisions – the selection of risk mitigations to perform. It computes the cost and benefit – cost is the cost of performing the selected risk mitigations, and benefit is the sum total of requirements attained.

TAR2, the summarization tool, takes as input a set of examples, each of which comprises a set of attribute values and an overall score. Attribute values must be drawn from finite, enumerable ranges (e.g., 1,2,3,4), as must scores. Furthermore, the possible values for score must be ordered. It outputs a set of alternative critical decision sets. Each critical decision set comprises ranges for a subset of the attribute values.

We interface these tools as follows:

- Each DDP mitigation becomes a separate TAR2 attribute, which can take on one of two values: "Y" or "N". "Y" corresponds to the mitigation being performed, "N" otherwise.
- For a given suite of mitigations to be performed, DDP automatically computes both a cost figure and a benefit figure. This computation is based on domain data. These two must be combined into a single score from an ordered set of possible such scores.
- TAR2 requires a *set* of examples. To generate such a set, run DDP repeatedly, each time randomly selecting mitigations to perform. In order to reach stable conclusions over large numbers of attributes, it may be necessary to generate a fairly large set of examples.
- TAR2, when provided a sufficiently large set of examples, returns several "treatments" – critical decision alternatives. Recall that a treatment consists of a subset of attribute range settings. In DDP terms, these can be mitigations to perform (TAR2 sets an attribute to the value "Y") and mitigations to *not* perform (TAR2 sets an attribute to the value "N"). Intuitively, the ones it recommends to perform are those essential to achieving a good score, while the ones it recommends to not perform are those detrimental to a good score.
- These critical decision alternatives are shown to the human experts. This gives them the opportunity to bring to bear additional knowledge that may not be encoded in the DDP model. For example, they may recognize an incompatibility between two of the mitigations that one of the alternatives recommends. They would therefore choose one of the other alternatives that TAR2 had recommended. They make a selection of one of the alternatives.
- The process is repeated, but with the mitigations set according to the critical decision selection made by the experts. Each iteration thus sets more and more of the mitigations, each one to either be performed or not.

5. Pilot Study

5.1. First iterative cycle

5.1.1 Generation of examples from requirements interaction model (DDP). The initial step of the pilot study involved generating a large number (30,000) of examples, sufficiently many for TAR2 to work with to reach stable conclusions. The black points on Figure 2 show the cost/benefit distribution of 10,000 of these 30,000 examples (if all 30,000 are plotted, the area becomes so densely filled that it is hard to discern the varying densities of points).

The pilot study needed an optimization target, e.g., maximize benefit for a given cost level. In the DDP model, cost is the dollar cost of performing the selected mitigations, while benefit is a measure of risk avoidance. The benefit scale is in arbitrary units of requirements weighting. The plot of initial values shows that at a cost of approximately $600,000 there are examples that attain near-maximal benefit of approximately 250. DDP applications typically seek near-maximal reduction of risk (i.e., near maximal benefit). This suggested an interesting and relevant challenge would be to seek to optimize at or around the cost limit of $600,000.

5.1.2 Combining cost and benefit values into a single score. DDP generates numerical cost and benefit values; TAR2 requires that these be combined to yield a single score, which can take on values from an enumerated range.

The target zone of costs at or below $600,000 motivated a partitioning of cost value into four regions: below \$600,000 (most desirable region); \$600,000 – \$649,999; \$650,000 – \$699,999; at or above \$700,000 (least desirable region). Benefits were partitioned by subdividing the data values into quartiles, i.e., putting the lowest 25% of the benefit figures into the lowest benefit range, the next 25% into the next, etc. Ranking the 16 possible pairings of cost and benefit then yielded a combined score of "goodness". During the pilot study we employed a mixture of two ways of combining cost and benefit scores. The first placed a greater priority on maximizing benefit, as shown in Table 1. Thus an example whose cost falls into the lowest cost range (i.e., up to \$600,000) and highest benefit range (top 25 percentile) would achieve the maximum possible score of ***16***. Next best, a score of ***15***, would go to examples in the 2nd cost range (\$600,000 – \$649,999) but still in the top 25 percentile benefit range, … the worst score, ***1***, is reserved for an example that falls into the highest cost, lowest benefit ranges. This ranking drove TAR2 towards high-benefit solutions.

Table 1 – benefit-prioritized score combination of cost & benefit

score		**COST RANGE**			
		1 (low)	**2**	**3**	**4 (high)**
BENEFIT RANGE	**4 (high)**	***16***	***15***	***14***	***13***
	3	***12***	***11***	***10***	***9***
	2	***8***	***7***	***6***	***5***
	1 (low)	***4***	***3***	***2***	***1***

The second combination scheme, shown in Table 2, stuck a more balanced combination of cost and benefit.

Table 2 – balanced score combination of cost & benefit

score		**COST RANGE**			
		1 (low)	**2**	**3**	**4 (high)**
BENEFIT RANGE	**4 (high)**	***16***	***14***	***11***	***7***
	3	***15***	***12***	***8***	***4***
	2	***13***	***9***	***5***	***2***
	1 (low)	***10***	***6***	***3***	***1***

The initial dataset was processed to assign the appropriate one of these scores to each of its 30,000 examples, so as to prepare it for TAR2.

5.1.3 Summarization using TAR2 treatment learning. TAR2 was applied to the processed dataset, and directed to look for treatment sets of increasingly large size. 6 was the maximum size for which it successfully terminated. The best treatment identified three mitigations to constrain to be performed, and three mitigations to constrain to *not* be performed.

To visualize the effect, DDP was then used to generate another large set of examples, with the mitigations in the TAR2 treatment constrained as indicated, and the remaining mitigations selected or not at random. Figures 3 and 4 (next page) show the distribution of the examples generated in the initial state, and the examples generated after the first iteration (10,000 examples are plotted for both cases). The improvements are dramatic – examples with low benefits (below 150) and/or high costs (over \$900,000) have vanished.

5.2. Successive iterations

In successive iterations, TAR2 was applied to the set of examples that emerged from executing the model on the best treatment found in the previous iteration. If this were a real-life application of the approach, after each of the iterations the experts would be presented the results so far, and asked to make their selection from among the multiple treatment sets proffered by TAR2.

The entire series is shown in Figures 3 through 8, on the next page. From the treatment sets it discovered, the best was selected, the model was additionally constrained by its recommendations, and another large set of examples generated. The stopping point, after the 5th iteration, is shown in Figure 8.

5.3. Stopping point

Following the 5th iteration, the variation among the benefit figures is relatively small. Recall that the underlying DDP data is human experts' estimates, and so the cost and benefit figures that DDP computes from these should not be misinterpreted to have high precision. Thus having reached the point where the benefit figures are so tightly clustered, it is appropriate to stop. A good strategy at this point is to pick one of the lower cost points along the upper border.

5.4. Sensitivity

Our requirements interaction model was populated with experts' estimates of the impact of risks on requirements, and effectiveness of mitigations at inhibiting risks. This has been the norm for DDP applications, targeted at problems for which historical data has rarely been available [4]. A question that often arises is to what extent the results suggested by DDP depend on the correctness of the experts' estimates. The availability of the TAR2 treatment learner gives us a way to study this issue.

We picked one of the near-optimal solutions that emerged from the iterations described above. This consisted of a set of mitigations to be applied. We then applied the same execution/summarization process upon this solution, but instead of varying which mitigations were selected for application while holding the model constant, we varied the model while holding the selection of mitigations constant. Each such experiment yielded a pair cost and benefit figures, but the cost figure remained constant (namely the sum total of the costs of the selected mitigations). The effects of varying the model showed up as changes to the computed benefit.

We generated 100,000 such randomly generated model variations, and fed these into TAR2 to search for model values (i.e., experts' estimates) that would lead to the greatest divergence from the originally computed benefit figure. TAR2 was unable to find any that made a significant difference. Based on TAR2's success in a number of domains, we are confident that critical settings must not exist, otherwise TAR2 would have found them. This strongly suggests that the recommendations found by our iterative process are not overly sensitive to variations in the experts' options contained within the model. That is, the recommendations would seem to be based on the aggregate effect of a large number of estimates, rather than critically dependant upon just a small subset of them.

6. Discussion

We have described a novel iterative process of execution, summarization and decision, the purpose of which is to converge towards near-optimal attainment of requirements in large-scale requirements models.

We claim to have demonstrated its success in our pilot study of a real-world instance of a requirements interaction model. In this section we defend this claim, and consider its broader implications, in particular, its applicability to other models.

6.1. Success on pilot study

The key claim we are making about the pilot study is that it showed success at arriving at a near-optimal attainment of requirements.

Without a baseline system to compare with, this claim cannot be rigorously defended. However, consider how the space of cost-benefits shrank as we iteratively applied TAR2. After five iterations, TAR2 had succeeded in identifying settings that demonstrably yield a compact set of points concentrated at the upper end of the benefit range, and at cost levels compatible with our initial goal of \$600,000. Indeed, the experiments revealed that almost as much benefit can be attained at somewhat lower costs, around the \$500,000 level. It is interesting to note that the combination of the treatments discovered by the five rounds of TAR2 constrained only one-third of the mitigations (30 out of 99), and yet these serve to narrow the spread of cost/benefit values significantly. The randomly generated mitigation suites at the start had benefits & costs dispersed widely across the 50 – 260 & \$300,000 – \$1,200,000 ranges. The stopping point has benefits & costs concentrated predominantly within the 240 – 265 & \$450,000 – \$650,000 ranges.

We are further encouraged by the sensitivity experiment, which indicated the near optimal solutions were robust with respect to the model inputs.

Figure 3. Initial state

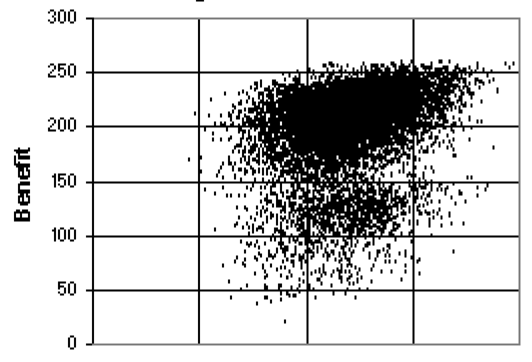


Figure 4. After 1st iteration

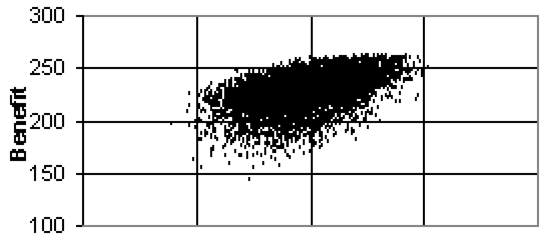


Figure 5. After 2nd iteration

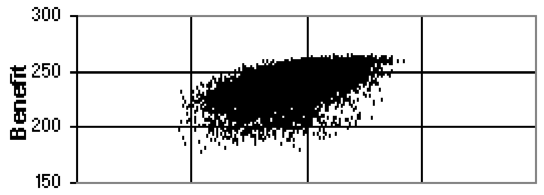


Figure 6. After 3rd iteration

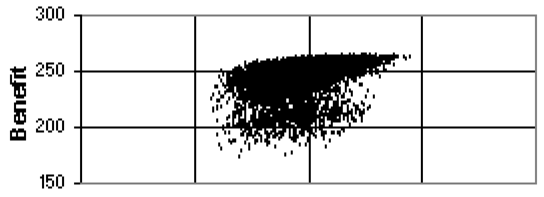


Figure 7. After 4th iteration

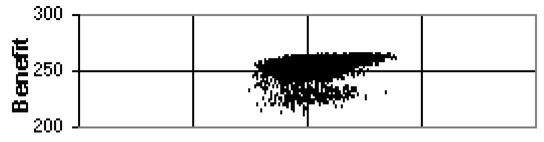
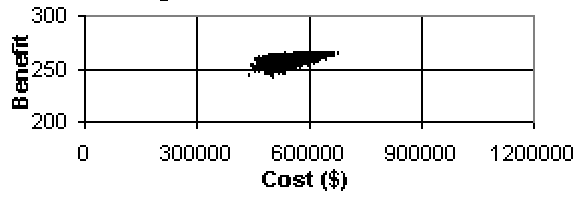


Figure 8. After 5th iteration



6.2. Novelty of the approach

The approach offers a way to converge to near-optimal attainment of requirements in large-scale requirements models. Furthermore, it identifies critical decision points along the way, giving human experts the opportunity to inject additional knowledge and guidance.

Potential alternative methods to our approach include traditional numeric optimization methods (e.g., linear programming) or computational intelligence methods (e.g., fuzzy logic, genetic algorithms, neural nets). However, numerical optimization cannot be applied here since DDP theories are discrete, not continuous. Numeric optimization is better suited to continuous theories containing smooth functions. Numeric methods are unsuited to discrete theories containing sudden “cliffs” (e.g. when a mitigation is activated).

Preliminary studies with a heuristic search technique based on use of genetic algorithms indicate it may provide a fast way of arriving at near-optimal solutions. However, each such solution takes the form of apply/don’t apply decisions for every mitigation, from which there is no apparent way to ascertain which of these are the most critical decisions. They drawbacks are lack of opportunity for experts to inject their additional knowledge into the process, and lack of focus on the critical decision points.

6.3. Wider applicability

We believe our iterative process to have applicability to a wide range of requirements interaction models. Our process requires that:

1. The requirements interaction model be “executable”, that is, can be randomly exercised to generate a set of examples, for each of which the model computes the measures of interest.
2. Those measures be combinable into a single measure that can take on values in a discrete, ordered range.
3. Summarization be capable of finding critical settings that lead towards the more desirable end of the ordered range.

In our case, it was easy to arrange to have DDP generate large numbers of examples, the only drawback being the time it took DDP to do so. At present, the DDP implementation on the large-scale pilot study’s dataset takes on the order of 3 hours to generate a set of 10,000 examples (running on a 1GHz machine with 1Gigabyte of RAM). The vast majority of the time goes into calculating the cost and benefit figures. One way around this would be to employ parallel processing, setting multiple CPUs the task of constructing examples by randomly selecting mitigations, and computing the cost and benefit figures for each example. We showed a simple scheme for abstracting them into ranges from which a single composite measure could be derived. These are easy steps for a wide range of models.

Note that the summarization component needs only examples generated by the requirements model, *not an understanding of the model itself.* Thus as the formalism used to capture requirements interaction models evolves, *no change to the summarization component is required*. We have been elaborating the DDP model, and so will be able to take advantage of this. It also means that the overall approach should work even if very different requirements interaction models are substituted.

We employed TAR2 for summarization. Given the lack of apparent alternatives, our belief in the wider applicability of this approach thus hinges on our belief in the wider applicability of TAR2. We address two areas of particular concern with this:

Is random generation of examples an adequate method for exploring a model? A theoretical drawback with any random search strategy is that such random exploration can miss significant parts of the space of options. A huge body of work testifies to the merits of random search, even for very hard tasks such as searching an argument space. For example, random search methods are very effective for scheduling problems and can solve hard and larger planning problems many times faster than traditional methods such as a systematic Davis-Putnam procedure [8]. A similar result was offered in [9] (this result is discussed below). That is, a random selection of mitigation strategies can be an adequate method for exploring an argument space.

Will the techniques described here scale to larger models? Our technique relies heavily on the TAR2 treatment leaner. Hence, our method won’t scale unless TAR2 also scales. Recall from section 3 that the algorithm explores all subsets of the “outstandingly promising attribute ranges”; i.e. those ranges that have most impact on changing the behavior of a system. Unless we can guarantee that the number of outstanding ranges is small, then this exponential search is intractable and our methods won’t scale. The “funnel theory” of [9] strongly suggests that only a small number of outstanding ranges will exist. The original funnel study tested relative effectiveness of exploring all/some resolutions to all/some arguments (where the “some” where randomly selected). In millions of runs, they observed a “funnel effect”; i.e. in most arguments, there exists a small set of key decisions that control all other decisions. [13] argued for the theoretical external validity of the funnel effect. They explored how random search might select between small funnels and large funnels. Based on known distributions of reaching part of a software system, they concluded that a random search is millions of times more likely to use small funnels. In systems with small funnels, a small number of decisions would be used frequently (i.e. those in the funnel). Those decisions would appear as outstandingly promising ranges, of which there would be only a few, and hence TAR2's exponential search would be tractable.

7. Conclusions

We have described a novel approach to converging upon near-optimal attainment of requirements in large-scale requirements models. It converges iteratively, at each stage identifying candidate sets of the most critical decisions that lead towards near-optimal solutions. This allows experts to inject their additional knowledge when selecting the decision set to use. In addition, this approach can also be used to investigate the sensitivity of the solution with respect to the requirements model's data.

We conducted a pilot study to investigate this approach. For this study we used a real-world requirements model of considerable size. The positive results of this study, together with our arguments for its wider applicability, suggest that this approach is worthy of further investigation. Now that we have this capability in place, we plan to make use of it to assist in future applications of our requirements modeling process. We also hope that this will stimulate additional interest in the use of this overall approach.

8. Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration, and at the University of British Columbia, Vancouver, Canada.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

Burton Sigal (JPL), John Kelly (JPL), Martha Wetherholt (NASA HQ), Tim Kurtz (NASA Glenn), Ken McGill (NASA IV&V) and Peter In (Texas A&M) have had a major and positive influence on this work. Steve Cornford (JPL) leads the entire DDP effort; he together with Julia Dunphy (JPL) and Jose Salcedo (JPL) have been studying the application of genetic algorithms as a way to locate near-optimal mitigation suites; their insights and contributions are gratefully acknowledged.

9. References

[1] B. Boehm, P. Bose, E. Horowitz & M. Lee. "Software Requirements as Negotiated Win Conditions", *Proceedings 1st International Conference on Requirements Engineering*, Colorado Springs, Colorado, 1994, pp 74-83.

[2] C.J. Burgess, I. Dattani, G. Hughes, J.H.R., May & K. Rees, "Using Influence Diagrams to Aid the Management of Software Change", *Requirements Engineering* 6(3), Oct 2001, pp 173-182.

[3] L. Chung, B.A. Nixon, E. Yu, J. Mylopoulos, *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishers, Boston, Oct 1999.

[4] S.L. Cornford, M.S. Feather & K.A. Hicks. "DDP – A tool for life-cycle risk management", *IEEE Aerospace Conference*, Big Sky, Montana, Mar 2001, pp 441-451.

[5] M.S. Feather, S.L. Cornford, M. Gibbel. "Scalable Mechanisms for Requirements Interaction Management", *Proceedings 4th IEEE International Conference on Requirements Engineering*, Schaumburg, Illinois, 19-23 Jun 2000, IEEE Computer Society, pp 119-129

[6] H. In, B. Boehm, T. Rodgers & M. Deutsch. "Applying WinWin to Quality Requirements: A Case Study", *Proceedings 23rd International Conference on Software Engineering*, Toronto, Ont., Canada, May 2001, pp 555-564.

[7] J. Karlsson & K. Ryan. "A Cost-Value Approach for Prioritizing Requirements", *IEEE Software*, Sept./Oct. 1997, pp 67-74.

[8], H. Kautz and B. Selman, Pushing the Envelope: Planning, Propositional Logic and Stochastic Search, *AAAI 96*, pp 1194-1201.

[9] T. Menzies, S. Easterbrook, B. Nuseibeh and S. Waugh. "An Empirical Investigation of Multiple Viewpoint Reasoning in Requirements Engineering", *Proceedings 4th IEEE International Symposium on Requirements Engineering*, 1999.

[10] T.J. Menzies and E. Sinsel. "Practical Large Scale What-if Queries: Case Studies with Software Risk Assessment", *Proceedings 15th IEEE International Conference on Automated Software Engineering*, Grenoble, France, Sept 2000, pp 165-173.

[11] T. Menzies & Y. Hu. "Constraining Discussions in Requirements Engineering via Models", *1st International Workshop on Model Based Requirements Engineering*, San Diego, California, Dec 2001.

[12] T. Menzies and J.D. Kiper. "Better reasoning about software engineering activities", *Proceedings 16th International Conference on Automated Software Engineering*, San Diego, California, Nov 2001, pp 391-394.

[13] T. Menzies and H. Singh. "Many Maybes Mean (Mostly) the Same Thing", *2nd International Workshop on Soft Computing applied to Software Engineering* (Netherlands), Feb, 2001.

[14] J. Mylopoulos, L. Chung, S. Liao, H. Wang & E. Yu. "Exploring Alternatives during Requirements Analysis", *IEEE Software* 18(1), Jan-Feb 2001, pp 92-96.

[15] R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufman, 1992.

[16] W.N. Robinson., S. Pawlowski & S. Volkov. "Requirements Interaction Management", GSU CIS Working Paper 99-7, Georgia State University, Atlanta, GA, August, 30, 1999.

[17] A. Tran & L. Chung. "NFR-Assistant: Tool Support for Achieving Quality", *Proceedings Application-Specific Systems and Software Engineering Technology*, Richardson, Texas, March 1999, pp 24-27.

[18] A. van Lamsweerde. "Goal-Oriented Requirements Engineering: A Guided Tour", *Proceedings 5th IEEE International Symposium on Requirements Engineering*, Toronto, Canada, August 2001, pp 249-263.