Measuring Inconsistency In Requirements Specifications

Kedian Mu[1], Zhi Jin[1,2], Ruqian Lu[1,2] and Weiru Liu[3]

[1] Institute of Computing Technology
Chinese Academy of Sciences, Beijing 100080, P.R.China
[2] MADIS, Academy of Mathematics and System Sciences
Chinese Academy of Sciences, Beijing 100080, P.R.China
[3] Department of Computer Science,
Queen's University Belfast, BT7 1NN, Northern Ireland

Abstract. In the field of requirements engineering, measuring inconsistency is crucial to effective inconsistency management. A practical measure must consider both the degree and significance of inconsistency in specification. The main contribution of this paper is providing an approach for measuring inconsistent specification in terms of the priority-based scoring vector, which integrates the measure of the degree of inconsistency with the measure of the significance of inconsistency. In detail, for each specification Δ that consists of a set of requirements statements, if L is a m-level priority set, we define a m-dimensional priority-based significance vector $\overrightarrow{V}$ to measure the significance of the inconsistency in Δ. Furthermore, a priority-based scoring vector $\overrightarrow{S_P}$: $\mathcal{P}(\Delta) \rightarrow \mathrm{N}^{m+1}$ has been defined to provide an ordering relation over specifications that describes which specification is "more essentially inconsistent than" others.

1 Introduction

It is widely recognized that inconsistency is unavoidable during the requirements stage, though most existing software development techniques or tools assume consistency [1–3]. A practical way of handling inconsistency is learning to live with inconsistency rather than parry it [3]. Furthermore, in many cases, it may be desirable to take the initiative in managing inconsistency to facilitate the requirements development and management [2]. Inconsistencies could be viewed as signals of problematical information about requirements.

Measuring inconsistency is crucial to effective inconsistency management [2, 1]. In general, customers and developers need to know the number and severity of inconsistencies in their requirements specifications. Often, developers need to use these measures to prioritize inconsistencies in order to identify inconsistencies that require urgent attentions, and to assess the progress after inconsistency-handling. In other words, the developers need to know if a set of requirements statements become more or less "consistent" after a particular inconsistency-handling action has been taken.

It is not surprising that techniques for measuring inconsistent specifications in classical logic are appealing [4]. In practical inconsistency-handling, customers and developers need to know both the significance and severity of inconsistency. The relative importance of a requirements statement always affects the evaluation of significance of an inconsistent specification. Therefore, central to measuring inconsistent specifications is the need to take the relative importance of requirements statements into account.

An approach to evaluating the significance of inconsistency in the framework of QC logic was proposed in [5] recently. It is based on specifying the relative significance of incoherent QC models using additional information, encoded as a mass assignment in Dempster-Shafer theory. This approach is not appropriate for measuring inconsistency in requirements specifications, though the QC logic is very appealing for representing inconsistent requirements specifications. Generally speaking, the relative importance of a requirements statement is implied by the relative priority of this statement in practical software development. But prioritization is just a strategy for differentiating requirements at a coarse granularity by relative importance and urgency. A common approach to prioritization is to group requirements statements into three priority categories, such as three-level scale of "*Essential*", "*Conditional*", and "*Optional*" [6, 7]. However, all such scales are subjective and imprecise, so it is difficult to specify the relative significance of inconsistency in the framework of Dempster-Shafer theory.

In this paper, we propose a new approach for measuring inconsistent specifications, which considers both the degree and significance of inconsistency based on the relative priorities of requirements statements. The rest of the paper is organized as follows. Section 2 introduces some preliminary notions. Section 3 presents the approach for measuring inconsistencies in requirements specifications. Finally, we conclude this paper in Section 4.

2 Preliminaries

As mentioned above, classical logic is appealing for representing the requirements specifications. We start this section with some notations of classical logic.

Let $\mathcal{L}_{\Phi_0}$ be the language composed from a set of classical atoms Φ_0 and logical connectives $\{\vee, \wedge, \neg, \rightarrow\}$ and let $\vdash$ be the classical consequence relation. Let $\alpha \in \mathcal{L}_{\Phi_0}$ be a classical formula and $\Delta \subseteq \mathcal{L}_{\Phi_0}$ a set of formulas in $\mathcal{L}_{\Phi_0}$. In this paper, we call Δ a requirements specification while each formula $\alpha \in \Delta$ represents a requirements statement.

Let $\mathsf{Consequence}(\Delta) = \{\alpha | \Delta \vdash \alpha\}$. If $\exists\alpha$ such that $\Delta \vdash \alpha$ and $\Delta \vdash \neg\alpha$, then we call Δ is inconsistent and abbreviate $\alpha \wedge \neg\alpha$ by $\bot$.

Generally, both the "plausible" and "problematical" information in the inconsistent set of formulas is of interest. However, for any set of formulas, we may consider each of its maximal consistent subsets as the reflection of one of many plausible views of the requirements specification. Furthermore, we consider the common subset of all its maximal consistent subsets as the reflection of all the "uncontroversial" information in it. On the other hand, we consider the union of

all its minimal inconsistent subsets as the collection of all the "problematical" information [8].

Definition 1. *Let Δ be a requirements specification. Then*

$$\mathit{CON}(\Delta) = \{\Gamma \subseteq \Delta | \Gamma \not\vdash \bot\},\ \mathit{INC}(\Delta) = \{\Gamma \subseteq \Delta | \Gamma \vdash \bot\}$$
$$\mathit{MC}(\Delta) = \{\Phi \in \mathit{CON}(\Delta) | \forall \Psi \in \mathit{CON}(\Delta), \Phi \not\subset \Psi\}$$
$$\mathit{MI}(\Delta) = \{\Phi \in \mathit{INC}(\Delta) | \forall \Psi \in \mathit{INC}(\Delta), \Psi \not\subset \Phi\}$$
$$\mathit{FREE}(\Delta) = \bigcap_{\Phi \in \mathit{MC}(\Delta)} \Phi = \Delta - \bigcup_{\Psi \in \mathit{MI}(\Delta)} \Psi,\ \mathit{CORE}(\Delta) = \Delta - \mathit{FREE}(\Delta)$$

$\mathsf{MC}(\Delta)$ is the set of maximal consistent subsets of Δ; $\mathsf{MI}(\Delta)$ is the set of minimal inconsistent subsets of Δ; and $\mathsf{FREE}(\Delta)$ is the set of formulas that appear in all the maximal consistent subsets of Δ.

Example 1. Let $\Delta = \{\alpha, \neg\gamma, \beta, \neg\beta \vee \gamma\}$, then $\mathsf{MC}(\Delta) = \{\Phi_1, \Phi_2, \Phi_3\}$,where $\Phi_1 = \{\alpha, \neg\gamma, \neg\beta \vee \gamma\}$, $\Phi_2 = \{\alpha, \beta, \neg\beta \vee \gamma\}$, and $\Phi_3 = \{\alpha, \neg\gamma, \beta\}$, $\mathsf{MI}(\Delta) = \{\{\neg\gamma, \beta, \neg\beta \vee \gamma\}\}$, and $\mathsf{FREE}(\Delta) = \{\alpha\}$.

For a set of formulas Δ, a scoring function S is defined from $\mathcal{P}(\Delta)$ (the power set of Δ) into the natural numbers so that for any $\Gamma \in \mathcal{P}(\Delta)$, $S(\Gamma)$ gives the number of minimal inconsistent subsets of Δ that would be eliminated if the subset Γ was removed from Δ [8]. That is, for $\Gamma \subseteq \Delta$, $S(\Gamma) = |\mathsf{MI}(\Delta)| - |\mathsf{MI}(\Delta - \Gamma)|$. As such, sets of formulas could be compared using their scoring functions so that an ordering relation, which means "*more inconsistent than*", over these sets can be defined.

Definition 2. *(**score ordering** [8], $\leq$) Assume that Δ_i and Δ_j are of the same cardinality, S_i is the scoring function for Δ_i, and S_j the scoring function for Δ_j. $S_i \leq S_j$ holds iff there is a bijection $f : \mathcal{P}(\Delta_i) \mapsto \mathcal{P}(\Delta_j)$ such that the following condition can be satisfied:*

$$\forall \Gamma \in \mathcal{P}(\Delta_i),\ S_i(\Gamma) \leq S_j(f(\Gamma))$$

*Note that $S_i < S_j$ iff $S_i \leq S_j$ and $S_j \not\leq S_i$. Also, $S_i \simeq S_j$ iff $S_i \leq S_j$ and $S_j \leq S_i$. We say Δ_j is **more inconsistent than** Δ_i iff $S_i \leq S_j$.*

3 Approach for Measuring Inconsistent Specification

Let m, a natural number, be the scale of the priority and L be $\{l_0^m, \cdots, l_{m-1}^m\}$, a totally ordered finite set of m symbolic values of the priorities, i.e. $l_i^m < l_j^m$ iff $i < j$. Furthermore, each symbolic value in L could associate with a linguistic value. For example, for a three-level priority set, we have a totally ordered set L as $L = \{l_0^3, l_1^3, l_2^3\}$ where [6, 7]

$$l_0^3 : \textit{Optional},\ l_1^3 : \textit{Conditional},\ l_2^3 : \textit{Essential}$$

In the rest of paper, we adopt this three-level priority set, though it is not obligatory. We may ignore the superscript m if no ambiguous arises. According to the convention in software engineering, the intuitive meaning of "*essential*" is that *the software product could not be acceptable unless all of the essential requirements are satisfied*; the meaning of "*conditional*" is that *these requirements would enhance the software product, but it is not unacceptable if absent*; the meaning of "*optional*" is that *these requirements may or may not be worthwhile.* In some sense, the priority could be seen as the abstraction of the requirements' significance.

Prioritizing requirements statements in Δ is in essence to establish a prioritizing mapping $P : \Delta \mapsto L$ by balancing the business benefit that each requirements statement can provide against its cost and technique risk.

Definition 3. *Let Δ be a requirements specification and L a m-level priority set. Let P be a prioritizing mapping $P : \Delta \mapsto L$. The priority-based partition of Δ under L can be defined as $\{\Delta^0, \cdots, \Delta^{m-1}\}$, such that*

$$\Delta^i = \{\alpha \in \Delta | P(\alpha) = l_i\}, \text{ for } i = 0, \cdots, m-1.$$

Obviously, each component of the priority-based partition of Δ is a subset of Δ. We give an example to illustrate the priority-based partition.

Example 2. Let L be a three-level priority set, and $\Delta = \{\alpha, \neg\gamma, \beta, \neg\beta \vee \gamma\}$. P is the prioritizing mapping from Δ to L:

$$P(\alpha) = l_2,\ P(\neg\gamma) = l_2,\ P(\beta) = l_1,\ P(\neg\beta \vee \gamma) = l_0$$

Then, $\Delta^0 = \{\neg\beta \vee \gamma\},\ \Delta^1 = \{\beta\},\ \Delta^2 = \{\alpha, \neg\gamma\}$. Obviously $\Delta = \Delta^0 \cup \Delta^1 \cup \Delta^2$.

For the priority-based partition of Δ under L, $\{\Delta^0, \cdots, \Delta^{m-1}\}$, Δ^i stands for the subset of Δ that is grouped to the category with priority level l_i. In other words, all of the requirements statements in Δ^i have the same level of relative importance and urgency. Note that, for Δ^l, the l-th component of its priority-based partition is itself, and others are $\emptyset$. For example, the priority-based partition of Δ^{m-1} is $\{\emptyset, \cdots, \emptyset, \Delta^{m-1}\}$.

3.1 Priority-based Score Ordering

Prioritizing requirements statements is in essence to differentiate the requirements statements by relative importance and urgency. In order to measure inconsistencies arising in requirements specifications, it is necessary to consider the relative priority of requirements statement in techniques. In fact, the approach based on scoring functions in [8] assumes that each formula has the same relative priority by default. In other words, it does not consider the significance of inconsistency. For the specifications consisting of requirements statements with different priorities as we have defined above, to consider their significance, we need to define a **priority-based score ordering** as follows to compare the inconsistent specifications.

Definition 4. *(**priority-based score ordering**, $\leq_P$) Let L be a m-level priority set. Let Δ_i and Δ_j be two specifications with the same cardinality. Let $\{\Delta_i^0, \cdots, \Delta_i^{m-1}\}$ and $\{\Delta_j^0, \cdots, \Delta_j^{m-1}\}$ be the priority-based partitions under L of Δ_i and Δ_j, respectively. Let S_i be the scoring function for Δ_i and S_j be the scoring function for Δ_j. $S_i \leq_P S_j$ holds iff there is a bijection $f : \mathcal{P}(\Delta_i) \mapsto \mathcal{P}(\Delta_j)$ such that the following conditions can be satisfied:*

- *f is a bijection from $\mathcal{P}(\Delta_i^l)$ to $\mathcal{P}(\Delta_j^l)$ ($l \in \{0, \cdots, m-1\}$);*
- *$\forall \Gamma \in \mathcal{P}(\Delta_i),\ S_i(\Gamma) \leq S_j(f(\Gamma))$*

*We call $\leq_P$ the priority-based score ordering. Note that $S_i <_P S_j$ iff $S_i \leq_P S_j$ and $S_j \not\leq_P S_i$, $S_i \simeq_P S_j$ iff $S_i \leq_P S_j$ and $S_j \leq_P S_i$. We say Δ_j is **more truly inconsistent** than Δ_i iff $S_i \leq_P S_j$.*

The priority-based score ordering emphasizes the bijection from $\mathcal{P}(\Delta_i^l)$ to $\mathcal{P}(\Delta_j^l)$, which provides a basis for comparing the scoring functions under the same level of priority. In other words, the significance of inconsistency is considered in the priority-based score ordering in an indirect way. Let us look at the following example to see how to compare inconsistent specifications via the priority-based score ordering.

Example 3. Let L be a three-level priority set. Consider $\Delta_1 = \{\neg\alpha, \alpha, \beta\}$ and $\Delta_2 = \{\alpha \wedge \neg\alpha, \beta, \gamma\}$. Assume that $\Delta_1^0 = \{\neg\alpha\}$, $\Delta_1^1 = \{\alpha\}$, $\Delta_1^2 = \{\beta\}$, $\Delta_2^0 = \{\alpha \wedge \neg\alpha\}$, $\Delta_2^1 = \{\beta\}$, and $\Delta_2^2 = \{\gamma\}$. Let S_1 and S_2 be the scoring functions for Δ_1 and Δ_2 respectively, as detailed below,

$$S_1(\{\neg\alpha\}) = 1,\ S_1(\{\alpha\}) = 1,\ S_1(\{\beta\}) = 0,\ S_1(\{\neg\alpha, \alpha\}) = 1$$
$$S_1(\{\neg\alpha, \beta\}) = 1,\ S_1(\{\alpha, \beta\}) = 1,\ S_1(\{\neg\alpha, \alpha, \beta\}) = 1$$
$$S_2(\{\alpha \wedge \neg\alpha\}) = 1,\ S_2(\{\beta\}) = 0,\ S_2(\{\gamma\}) = 0,\ S_2(\{\alpha \wedge \neg\alpha, \beta\}) = 1$$
$$S_2(\{\alpha \wedge \neg\alpha, \gamma\}) = 1,\ S_2(\beta, \gamma\}) = 0,\ S_2(\{\alpha \wedge \neg\alpha, \beta, \gamma\}) = 1$$

Then we have $S_2 < S_1$. Therefore, if we ignore the relative significance of each formula in $\Delta_1 \cup \Delta_2$, we conclude that Δ_1 is *more inconsistent than* Δ_2. But if we consider the relative significance of each formula, then $S_2 <_P S_1$. We may say that Δ_1 is *more truly inconsistent than* Δ_2.

The priority-based score ordering considers both the degree and significance of inconsistency in some sense. It is more strict than the score ordering. That could be shown by the following proposition.

Proposition 1. *Let Δ_i and Δ_j be of the same cardinality. If S_i and S_j are the scoring functions for Δ_i and Δ_j respectively, then $S_i \leq_P S_j$ implies $S_i \leq S_j$. But the converse does not hold.*

Example 4. (a counterexample for the converse) Let L be a three-level priority set. Consider $\Delta_1 = \{\neg\alpha, \alpha, \beta\}$ and $\Delta_2 = \{\alpha \wedge \neg\alpha, \beta, \gamma\}$. Assume that $\Delta_1^0 = \{\neg\alpha, \alpha\}$, $\Delta_1^2 = \{\beta\}$, $\Delta_2^0 = \{\alpha \wedge \neg\alpha\}$, $\Delta_2^1 = \{\beta\}$, and $\Delta_2^2 = \{\gamma\}$. There is no bijection from $\mathcal{P}(\Delta_1^0)$ to $\mathcal{P}(\Delta_2^0)$, thus $S_2 < S_1$ but $S_2 \not<_P S_1$.

3.2 Measuring Significance of Inconsistent Specification

The priority-based score ordering does not provide a direct approach for measuring the significance of inconsistency based on the priority. It just provides a basis for comparing the scoring functions under the same level of priority. As mentioned above, the priority associated with each requirements statement is some kind of abstraction of this statement's significance. We may easily think up the following intuitive assumptions: (1) the requirements statements with the same priority have the same significance; (2) any requirements statement with higher priority is more significant than all of those with lower priorities; (3) those requirements statements with higher priorities play dominant roles in measuring the significance of the inconsistencies in requirements specifications. This is the reason why we have to take the priority into account. To achieve this objective, we first introduce a priority-based cardinality vector for Δ.

Definition 5. *Let L be a m-level priority set. $\forall \Delta \subseteq \mathcal{L}_{\Phi_0}$, the priority-based cardinality vector of Δ, denoted $\overrightarrow{C}(\Delta)$, is defined as $\overrightarrow{C}(\Delta) = (|\Delta^0|, \cdots, |\Delta^{m-1}|)$, where $\{\Delta^0, \cdots, \Delta^{m-1}\}$ is the priority-based partition of Δ under L and $|\Delta^l|$ is cardinality of Δ^l for each $l \in \{0, \cdots, m-1\}$.*

Example 5. Consider $\Delta = \{\alpha, \beta, \neg\beta \vee \neg\alpha, \gamma\}$. Let L be a three-level priority set. Let $\{\Delta^0, \Delta^1, \Delta^2\}$ be the priority-based partition of Δ under L, where $\Delta^0 = \{\beta, \neg\beta \vee \neg\alpha\}$, $\Delta^1 = \{\alpha\}$, and $\Delta^2 = \{\gamma\}$, then $\overrightarrow{C}(\Delta) = (2, 1, 1)$.

Definition 6. *(**cardinality vector ordering**, $\preceq_P$) Let $\Delta \subseteq \mathcal{L}_{\Phi_0}$, $\Gamma_i, \Gamma_j \subseteq \Delta$, and L a m-level priority set. Let $\overrightarrow{C}(\Gamma_i)$ and $\overrightarrow{C}(\Gamma_j)$ be the priority-based cardinality vectors under L of Γ_i and Γ_j respectively. The cardinality vector ordering, denoted $\preceq_P$, is defined as: $\overrightarrow{C}(\Gamma_i) \preceq_P \overrightarrow{C}(\Gamma_j)$ iff $\exists 0 \leq l \leq m-1$ such that $|\Gamma_i^l| \leq |\Gamma_j^l|$ and $\forall l < k \leq m-1, |\Gamma_i^k| = |\Gamma_j^k|$. Furthermore, $\overrightarrow{C}(\Gamma_i) \prec_P \overrightarrow{C}(\Gamma_j)$ iff $\overrightarrow{C}(\Gamma_i) \preceq_P \overrightarrow{C}(\Gamma_j)$ and $\overrightarrow{C}(\Gamma_j) \not\preceq_P \overrightarrow{C}(\Gamma_i)$; $\overrightarrow{C}(\Gamma_i) = \overrightarrow{C}(\Gamma_j)$ iff $\overrightarrow{C}(\Gamma_i) \preceq_P \overrightarrow{C}(\Gamma_j)$ and $\overrightarrow{C}(\Gamma_j) \preceq_P \overrightarrow{C}(\Gamma_i)$.*

In this sense, the priority-based cardinality vector $\overrightarrow{C}(\Delta)$ gives a measure of priority-based significance of Δ. The l-th component of $\overrightarrow{C}(\Delta)$ denote the number of the requirements with the l-th level of priority.

Proposition 2. *Let L be a m-level priority set. Let $\Delta \subseteq \mathcal{L}_{\Phi_0}$. If $\overrightarrow{C}$ denotes the priority-based cardinality vector under L, then for $\Gamma_i,\ \Gamma_j \subseteq \Delta$,*

$$\overrightarrow{C}(\Gamma_i \cap \Gamma_j) \preceq_P \min_{\preceq_P}(\overrightarrow{C}(\Gamma_\mathrm{i}), \overrightarrow{C}(\Gamma_\mathrm{j}))$$
$$\max_{\preceq_P}(\overrightarrow{C}(\Gamma_\mathrm{i}), \overrightarrow{C}(\Gamma_\mathrm{j})) \preceq_P \overrightarrow{C}(\Gamma_\mathrm{i} \cup \Gamma_\mathrm{j})$$

where $min_{\preceq_P}(\overrightarrow{C}(\Gamma_i), \overrightarrow{C}(\Gamma_j)) = \overrightarrow{C}(\Gamma_i)$ if $\overrightarrow{C}(\Gamma_i) \preceq_P \overrightarrow{C}(\Gamma_j)$, or $\overrightarrow{C}(\Gamma_j)$ otherwise; $max_{\preceq_P}(\overrightarrow{C}(\Gamma_i), \overrightarrow{C}(\Gamma_j)) = \overrightarrow{C}(\Gamma_j)$ if $\overrightarrow{C}(\Gamma_i) \preceq_P \overrightarrow{C}(\Gamma_j)$, or $\overrightarrow{C}(\Gamma_i)$ otherwise.

Now we can use the priority-based cardinality vector to describe the significance of inconsistency. Let N be a set of natural numbers, then $\mathrm{N^m}$ is a m-dimensional space.

Definition 7. *Let L be a m-level priority set and $\Delta \subseteq \mathcal{L}_{\Phi_0}$. The priority-based significance vector for Δ under L, $\overrightarrow{V} : \mathcal{P}(\Delta) \mapsto \mathrm{N^m}$, can be defined as that for $\Gamma \in \mathcal{P}(\Delta)$,*

$$\overrightarrow{V}(\Gamma) = \overrightarrow{C}(\mathsf{CORE}(\Delta)) - \overrightarrow{C}(\mathsf{CORE}(\Delta - \Gamma))$$

If we use $V^l(\Gamma)$ to denote $|\mathsf{CORE}(\Delta)^l| - |\mathsf{CORE}(\Delta - \Gamma)^l|$ for each $l \in \{0, \cdots, m-1\}$, then $\overrightarrow{V}(\Gamma) = (V^0(\Gamma), \cdots, V^{m-1}(\Gamma))$.

Intuitively, for $\Gamma \in \mathcal{P}(\Delta)$, $\overrightarrow{V}(\Gamma)$ captures the reduction of the significance of those "problematical" statements in Δ after Γ were removed from Δ. Based on $\overrightarrow{V}$, we may introduce another ordering relation, the *priority-based significance ordering*, for comparing the significance of inconsistencies in specifications.

Definition 8. ***(priority-based significance ordering**, $\preceq_P^S$) Let L be a m-level priority set. Assume that Δ_i and Δ_j are of the same cardinality. Let $\overrightarrow{V_i}$ and $\overrightarrow{V_j}$ be the priority-based significance vectors under L for Δ_i and Δ_j respectively. Then $\overrightarrow{V_i} \preceq_P^S \overrightarrow{V_j}$ holds iff there is a bijection $f : \mathcal{P}(\Delta_i) \mapsto \mathcal{P}(\Delta_j)$ such that the following condition can be satisfied:*

$$\forall \Gamma \in \mathcal{P}(\Delta_i), \overrightarrow{V_i}(\Gamma) \preceq_P \overrightarrow{V_j}(f(\Gamma))$$

*We call $\preceq_P^S$ the priority-based significance ordering. Furthermore, $\overrightarrow{V_i} \prec_P^S \overrightarrow{V_j}$ iff $\overrightarrow{V_i} \preceq_P^S \overrightarrow{V_j}$ and $\overrightarrow{V_j} \not\preceq_P^S \overrightarrow{V_i}$; $\overrightarrow{V_i} \simeq_P^S \overrightarrow{V_j}$ iff $\overrightarrow{V_i} \preceq_P^S \overrightarrow{V_j}$ and $\overrightarrow{V_j} \preceq_P^S \overrightarrow{V_i}$. We say the inconsistency in Δ_j is **more significant than** that in Δ_i iff $\overrightarrow{V_i} \preceq_P^S \overrightarrow{V_j}$.*

Let us give an example to illustrate how to compare two inconsistent specifications from the significance of inconsistency via the priority-based significance ordering.

Example 6. Consider $\Delta_1 = \{\alpha, \neg\alpha\}$ and $\Delta_2 = \{\beta, \neg\beta\}$. Let L be a three-level priority set. Assume that $\Delta_1^0 = \{\alpha\}$, $\Delta_1^1 = \{\neg\alpha\}$, $\Delta_2^2 = \{\beta\}$, and $\Delta_2^2 = \{\neg\beta\}$. If $\overrightarrow{V_1}$ and $\overrightarrow{V_2}$ are priority-based significance vectors under L for Δ_1 and Δ_2 respectively, then

$$\overrightarrow{V_1}(\Delta_1) = (1,1,0), \overrightarrow{V_1}(\{\alpha\}) = (1,1,0), \overrightarrow{V_1}(\{\neg\alpha\}) = (1,1,0)$$
$$\overrightarrow{V_2}(\Delta_2) = (0,1,1), \overrightarrow{V_2}(\{\beta\}) = (0,1,1), \overrightarrow{V_2}(\{\neg\beta\}) = (0,1,1)$$

Therefore, $\overrightarrow{V_1} \prec_P^S \overrightarrow{V_2}$, and we conclude that the inconsistency in Δ_2 is **more significant than** that in Δ_1. However, if we apply the scoring function, S, to Δ_1 and Δ_2, we can not tell the difference of their inconsistencies.

Proposition 3. *Let L be a m-level priority set. Let $\Delta_i,\ \Delta_j \subseteq \mathcal{L}_{\Phi_0}$. If $\overrightarrow{V_i}$ and $\overrightarrow{V_j}$ are the priority-based significance vectors under L for Δ_i and Δ_j respectively, then $\overrightarrow{V_i} \preceq_P^S \overrightarrow{V_j}$ implies $\overrightarrow{C}(\mathsf{CORE}(\Delta_i)) \preceq_P \overrightarrow{C}(\mathsf{CORE}(\Delta_j))$.*

The priority-based significance vector provides a concise means for articulating the significance of inconsistency in specifications. For inconsistent specifications, it is easy to get the following relation between the degree and significance of inconsistency.

Proposition 4. *Let L be a m-level priority set and $\Delta \subseteq \mathcal{L}_{\Phi_0}$. Let $\overrightarrow{0}$ be a zero vector. If S is the scoring function for Δ and $\overrightarrow{V}$ the priority-based significance vector for Δ under L, then for $\Gamma \subseteq \Delta$, $S(\Gamma) = 0$ iff $\overrightarrow{V}(\Gamma) = \overrightarrow{0}$.*

3.3 Priority-based Scoring Vector

As mentioned earlier, the scoring function S for Δ reveals the degree of inconsistency arising in Δ, while the priority-based significance vector $\overrightarrow{V}$ for Δ measures the significance of inconsistency. We also give two ordering relations for comparing inconsistent specifications from the perspectives of the degree and the significance of inconsistency, respectively. Actually, in many cases, we need to consider both of them. In software engineering, we might define this integrated measure by combining the scoring function with the priority-based significance vector.

Definition 9. *Let L be a m-level priority set and $\Delta \subseteq \mathcal{L}_{\Phi_0}$. Let $\overrightarrow{V}$ be the priority-based significance vectors under L for Δ. The priority-based scoring vector for Δ under L, $\overrightarrow{S_P}$: $\mathcal{P}(\Delta) \mapsto \mathrm{N}^{\mathrm{m+1}}$, can be defined as that for $\Gamma \in \mathcal{P}(\Delta)$,*

$$\overrightarrow{S_P}(\Gamma) = (V^0(\Gamma), \cdots, V^{m-1}(\Gamma), S(\Gamma))$$

Actually, for $\Gamma \in \mathcal{P}(\Delta)$, the priority-based scoring vector for Δ consists of $\overrightarrow{V}(\Gamma)$ concatenated with value $S(\Gamma)$. It could be viewed as the integrated measure of inconsistent information of Δ that would be reduced if Γ were removed from Δ. Furthermore, we can compare these inconsistent specifications using the priority-based scoring vector for each specification from an integrated view.

Definition 10. *(**scoring vector ordering**, $\preceq_P^*$) Let $\Delta \subseteq \mathcal{L}_{\Phi_0}$, $\Gamma_i, \Gamma_j \subseteq \Delta$, and L a m-level priority set. Let $\overrightarrow{S_P}(\Gamma_i)$ and $\overrightarrow{S_P}(\Gamma_j)$ be the priority-based scoring vectors under L of Γ_i and Γ_j respectively. The scoring vector ordering, denoted $\preceq_P^*$, is defined as: $\overrightarrow{S_P}(\Gamma_i) \preceq_P^* \overrightarrow{S_P}(\Gamma_j)$ iff $\overrightarrow{V}(\Gamma_i) \preceq_P \overrightarrow{V}(\Gamma_j)$ and $S(\Gamma_i) \leq S(\Gamma_j)$. Furthermore, $\overrightarrow{S_P}(\Gamma_i) \prec_P^* \overrightarrow{S_P}(\Gamma_j)$ iff $\overrightarrow{S_P}(\Gamma_i) \preceq_P^* \overrightarrow{S_P}(\Gamma_j)$ and $\overrightarrow{S_P}(\Gamma_j) \not\preceq_P^* \overrightarrow{S_P}(\Gamma_i)$; $\overrightarrow{S_P}(\Gamma_i) = \overrightarrow{S_P}(\Gamma_j)$ iff $\overrightarrow{S_P}(\Gamma_i) \preceq_P^* \overrightarrow{S_P}(\Gamma_j)$ and $\overrightarrow{S_P}(\Gamma_j) \preceq_P^* \overrightarrow{S_P}(\Gamma_i)$;*

Definition 11. *(**priority-based score vector ordering**, $\preceq_P^E$) Let L be a m-level priority set. Assume that Δ_i and Δ_j are of the same cardinality. Let $\overrightarrow{S_{Pi}}$ and $\overrightarrow{S_{Pj}}$ be the priority-based scoring vectors under L for Δ_i and Δ_j, respectively. $\overrightarrow{S_{Pi}} \preceq_P^E \overrightarrow{S_{Pj}}$ holds iff there is a bijection $f : \mathcal{P}(\Delta_i) \mapsto \mathcal{P}(\Delta_j)$ such that the following condition can be satisfied: $\forall \Gamma \in \mathcal{P}(\Delta_i)$, $\overrightarrow{S_{Pi}}(\Gamma) \preceq_P^* \overrightarrow{S_{Pj}}(f(\Gamma))$. We call $\preceq_P^E$ the priority-based score vector ordering. Furthermore, $\overrightarrow{S_{Pi}} \prec_P^E \overrightarrow{S_{Pj}}$ iff $\overrightarrow{S_{Pi}} \preceq_P^E \overrightarrow{S_{Pj}}$ and $\overrightarrow{S_{Pj}} \not\preceq_P^E \overrightarrow{S_{Pi}}$; $\overrightarrow{S_{Pi}} \simeq_P^E \overrightarrow{S_{Pj}}$ iff $\overrightarrow{S_{Pi}} \preceq_P^E \overrightarrow{S_{Pj}}$ and $\overrightarrow{S_{Pj}} \preceq_P^E \overrightarrow{S_{Pi}}$. We say Δ_j is **more essentially inconsistent** than Δ_i iff $\overrightarrow{S_{Pi}} \preceq_P^E \overrightarrow{S_{Pj}}$.*

Proposition 5. *Let L be a m-level priority set, and Δ_i, $\Delta_j \subseteq \mathcal{L}_{\Phi_0}$. Let S_i and S_j be the scoring functions for Δ_i and Δ_j respectively. If $\overrightarrow{S_{Pi}}$ and $\overrightarrow{S_{Pj}}$ are the priority-based scoring vectors under L for Δ_i and Δ_j respectively, then $\overrightarrow{S_{Pi}} \preceq_P^E \overrightarrow{S_{Pj}}$ implies $\overrightarrow{V_i} \preceq_P^S \overrightarrow{V_j}$ and $S_i \leq S_j$.*

Let us look at the following example to see how to compare two inconsistent specifications from two different perspectives, i.e. the degree and the significance of inconsistency.

Example 7. Consider $\Delta_1 = \{\alpha, \beta, \neg\alpha, \neg\beta\}$ and $\Delta_2 = \{\alpha, \gamma, \neg\alpha, \neg\gamma\}$. Let L be a three-level priority set. And let $\{\Delta_1^0, \Delta_1^1, \Delta_1^2\}$ and $\{\Delta_2^0, \Delta_2^1, \Delta_2^2\}$ be the priority-based partitions under L of Δ_1 and Δ_2, respectively, where $\Delta_1^0 = \{\alpha\}$, $\Delta_1^1 = \{\neg\beta\}$, $\Delta_1^2 = \{\neg\alpha, \beta\}$, $\Delta_2^0 = \{\alpha, \neg\alpha\}$, $\Delta_2^1 = \{\neg\gamma\}$, and $\Delta_2^2 = \{\gamma\}$.

If S_1 and S_2 are the scoring functions for Δ_1 and Δ_2 respectively, then $S_1 \simeq S_2$. Therefore, we may say Δ_1 is ***as inconsistent as*** Δ_2 from the perspective of the ***degree of inconsistency***. On the other hand, from the perspective of the ***significance of inconsistency***, we may say the inconsistency in Δ_1 is ***more significant than*** that in Δ_2 since $\overrightarrow{V_2} \prec_P^S \overrightarrow{V_1}$, where $\overrightarrow{V_1}$ and $\overrightarrow{V_2}$ are the priority-based significance vectors under L for Δ_1 and Δ_2 respectively. Furthermore, if $\overrightarrow{S_{P1}}$ and $\overrightarrow{S_{P2}}$ are the priority-based scoring vectors under L for Δ_1 and Δ_2 respectively, we have $\overrightarrow{S_{P2}} \prec_P^E \overrightarrow{S_{P1}}$. That is, from the integrative perspective, Δ_1 is ***more essentially inconsistent than*** Δ_2.

However, as illustrated by the following propositions, the priority-based scoring vector is also a concise and yet expressive articulation of the inconsistencies that arise in requirements specifications from both the severity and significance.

Proposition 6. *Let $\Delta \subseteq \mathcal{L}_{\Phi_0}$ and L a m-level priority set. If $\overrightarrow{S_P}$ is the priority-based scoring vector under L for Δ, then*

$$\overrightarrow{S_P}(\mathsf{FREE}(\Delta)) = \overrightarrow{0}, \overrightarrow{S_P}(\mathsf{CORE}(\Delta)) = (|\mathsf{CORE}(\Delta)^0|, \cdots, |\mathsf{CORE}(\Delta)^{m-1}|, |\mathsf{MI}(\Delta)|);$$

Proposition 7. *Let $\Delta \subseteq \mathcal{L}_{\Phi_0}$ and L a m-level priority set. If $\overrightarrow{S_P}$ is the priority-based scoring vector under L for Δ, then for $\alpha \in \Delta$,*

$$\alpha \in \mathsf{FREE}(\Delta) \text{ iff } \overrightarrow{S_P}(\{\alpha\}) = \overrightarrow{0}; \text{ and } \alpha \in \mathsf{CORE}(\Delta) \text{ iff } \overrightarrow{0} \prec_P^* \overrightarrow{S_P}(\{\alpha\})$$

Proposition 8. *Let $\Delta \subseteq \mathcal{L}_{\Phi_0}$ and L a m-level priority set. If $\overrightarrow{S_P}$ is the priority-based scoring vector under L for Δ, then for $\Gamma_i,\ \Gamma_j \subseteq \Delta$,*

$$\overrightarrow{S_P}(\Gamma_i \cap \Gamma_j) \preceq_P^* min_{\preceq_P^*}(\overrightarrow{S_P}(\Gamma_i), \overrightarrow{S_P}(\Gamma_j))$$
$$max_{\preceq_P^*}(\overrightarrow{S_P}(\Gamma_i), \overrightarrow{S_P}(\Gamma_j)) \preceq_P^* \overrightarrow{S_P}(\Gamma_i \cup \Gamma_j)$$

where $min_{\preceq_P^*}(\overrightarrow{S_P}(\Gamma_i), \overrightarrow{S_P}(\Gamma_j)) = \overrightarrow{S_P}(\Gamma_i)$ if $\overrightarrow{S_P}(\Gamma_i) \preceq_P^* \overrightarrow{S_P}(\Gamma_j)$, or $\overrightarrow{S_P}(\Gamma_j)$ otherwise; $max_{\preceq_P^*}(\overrightarrow{S_P}(\Gamma_i), \overrightarrow{S_P}(\Gamma_j)) = \overrightarrow{S_P}(\Gamma_j)$ if $\overrightarrow{S_P}(\Gamma_i) \preceq_P^* \overrightarrow{S_P}(\Gamma_j)$, or $\overrightarrow{S_P}(\Gamma_i)$ otherwise.

Proposition 9. *Let L be a m-level priority set and $\Delta_i,\ \Delta_j \subseteq \mathcal{L}_{\Phi_0}$. If $\overrightarrow{S_{Pi}}$ and $\overrightarrow{S_{Pj}}$ are the priority-based scoring vectors under L for Δ_i and Δ_j respectively, then $\overrightarrow{S_{Pi}} \preceq_P^E \overrightarrow{S_{Pj}}$ implies $|\mathsf{FREE}(\Delta_i)| \geq |\mathsf{FREE}(\Delta_j)|$. But the converse does not hold.*

Example 8. (a counterexample for the converse). Consider $\Delta_1 = \{\alpha, \neg\alpha, \beta\}$ and $\Delta_2 = \{\alpha \wedge \neg\alpha, \beta, \gamma\}$. Let L be a three-level priority set. Let $\Delta_1^0 = \{\alpha, \neg\alpha\}$, $\Delta_1^2 = \{\beta\}$, $\Delta_2^1 = \{\beta, \gamma\}$, and $\Delta_2^2 = \{\alpha \wedge \neg\alpha\}$. So, $|\mathsf{FREE}(\Delta_2)| > |\mathsf{FREE}(\Delta_1)|$, but $\overrightarrow{S_{P2}} \not\preceq_P^E \overrightarrow{S_{P1}}$

3.4 Case Study

Example 9. Let L be a three-level priority set. Consider a scenario in a close residential area management system. Developer A, who is in charge of gathering information about vehicle's application for entrance, supplies the "*essential*" requirements as follows: *The vehicles with authorization (Auth) of residential area can enter (Enter) the area; The vehicles without authorization can not enter.* He also gathers a legal rule about *fire engine* as follows: *the fire engine (Fire) can enter the area without authorization.* If we use Δ_A to represent the specification from A, then Δ_A contains:

$$Fire(v) \rightarrow Enter(v), Fire(v) \rightarrow \neg Auth(v), \neg Auth(v) \rightarrow \neg Enter(v), Fire(v)$$

The priority-based partition of Δ_A is: $\Delta_A^2 = \Delta_A$.

Developer B, who is in charge of managing renting garages, supplies the "*essential*" requirements as follows: *A garage is available (Available) if it is unoccupied (Unoccupied).* A further "*conditional*" requirements is: *If a garage should be repaired (Repaired), then it is not available; If a garage can be repaired, then it is unoccupied.* Then specification Δ_B contains the following statements:

$$Unoccupied(a) \rightarrow Available(a),\ Repaired(a) \rightarrow \neg Available(a),$$
$$Repaired(a) \rightarrow Unoccupied(a),\ Repaired(a)$$

The priority-based partition of Δ_B is: $\Delta_B^1 = \{Repaired(a) \rightarrow \neg Available(a), Repaired(a) \rightarrow Unoccupied(a)\}, \Delta_B^2 = \Delta_B - \Delta_B^1$. Obviously, both Δ_A and Δ_B are inconsistent. If $\overrightarrow{S_{PA}}$ and $\overrightarrow{S_{PB}}$ are the priority-based scoring vectors under L of Δ_A and Δ_B, respectively, then $\overrightarrow{S_{PB}} \prec_P^E \overrightarrow{S_{PA}}$. It signifies that the developers should give Δ_A priority based on integrated measure of inconsistency. However, if we use the scoring functions in [8], we can't distinguish the inconsistencies of the two specifications.

The approach could also be applied to other scenarios such as negotiation between agents and the comparison of heterogeneous sources of information, since the relative importance of knowledge in certain scenario may affect the measure of inconsistency, especially in competitive negotiation.

Example 10. Consider the competition of Japan and China for Russia's oil and gas pipeline routes. Generally, large amount of the export of oil, dominant role in export, the length and cost of routes are viewed as factors that may contribute to Russia's choice of routes.

Let Δ_R be Russia's perspective about routes. $\Delta_R = \{short, cheap, large, dominant\}$.

Let the descriptions of routes proposed by China and Japan be represented by Δ_C and Δ_J respectively. $\Delta_C = \{short,\ cheap,\ \neg large,\ \neg dominant\}$, $\Delta_J = \{\neg short,\ \neg cheap,\ large,\ dominant\}$.

Hence, the negotiation between Russia and China is captured by Δ_{RC}.

$$\Delta_{RC} = \{short,\ cheap,\ \neg large,\ \neg dominant,\ large,\ dominant\}$$

The negotiation between Russia and Japan is captured by Δ_{RJ}.

$$\Delta_{RJ} = \{short,\ cheap,\ large,\ dominant,\ \neg short,\ \neg cheap\}$$

Let L be a three-level priority set. As for the items that contribute to Russia's choice of routes, large amount of the export of oil and dominant role in export are essential factors, while the length and cost of route are significant but less essential factors. Therefore, the priority-based partition of Δ_{RC} is captured as follows:

$$\Delta^1_{RC} = \{short,\ cheap\}, \Delta^2_{RC} = \{\neg large,\ \neg dominant,\ large,\ dominant\}$$

The priority-based partition of Δ_{RJ} is captured as follows:

$$\Delta^1_{RJ} = \{short,\ cheap,\ \neg short,\ \neg cheap\}, \Delta^2_{RJ} = \{large,\ dominant\}$$

If $\overrightarrow{S_{PRC}}$ is the priority-based scoring vector for Δ_{RC} and $\overrightarrow{S_{PRJ}}$ is the priority-based scoring vector for Δ_{RJ}, then $\overrightarrow{S_{PRJ}} \prec^E_P \overrightarrow{S_{PRC}}$. It implies that Japanese proposal of pipeline route is more attractive to Russia than that of China.

4 Conclusions

In terms of the relative priorities of requirements statements, this paper presents a set of priority-based strategies to measure the inconsistencies arising in requirements specifications. First, the priority-based score ordering is proposed to compare the degree of inconsistencies under the same level of priority. And then the priority-based significance vector is given to assess the significance of inconsistency. And finally, the priority-based score vector ordering, which is based on the priority-based scoring vector, is defined to compare the inconsistent specifications from an integrated view, i.e. according to both the degree and the significance of inconsistency.

Measuring inconsistency is still an important issue in developing requirements specifications as well as intelligent systems. Some recent techniques for measuring inconsistent information have been reviewed in [9]. The overwhelming majority of these techniques focus on different measures of the degree of inconsistency [10–13]. At present, the scoring function [8] is one of the most appropriate tools for summarizing the degree of inconsistency. However, researchers have begun to study the significance of inconsistency. For example, Hunter provided a approach for measuring the significance of inconsistency arising in QC models [5]. This approach is based on specifying the relative significance of incoherent models using additional information, encoded as a mass assignment. But, the priority of a requirements statement is just an imprecise measure of relative importance. It is difficult to determine the precise measure of relative significance for each statement during the requirements stage in many cases. That might be the main obstacles in putting this approach into practical applications.

In contrast, the approach described in this paper uses the priority-based significance vector to measure the significance of inconsistency. The priority-based partition of specification is available during the requirements stage [14].

It could be viewed as a partition of requirements by relative importance and urgency. Moreover, in general cases, the priority-based partition of specification is accepted by all stakeholders. That is, each stakeholder gives the same meaning of the same level of significance. It shows that this approach may be more feasible to requirements engineering practices.

Acknowledgements

This work was partly supported by the National Natural Science Foundation of China (No.60233010 and No.60496324), the National Key Research and Development Program (Grant No. 2002CB312004) of China, the Knowledge Innovation Program of the Chinese Academy of Sciences and the British Royal Society China-UK Joint Project. We are grateful to the reviewers for their constructive comments, which helped to improve our work.

References

1. Nuseibeh, B., Easterbrook, S., Russo, A.: Leveraging inconsistency in software development. IEEE Computer **33** (2000) 24–29
2. Nuseibeh, B., S.Easterbrook, A.Russo: Making inconsistency respectable in software development. Journal of Systems and Software **58** (2001) 171–180
3. Easterbrook, S., M.Chechik: 2nd international workshop on living with inconsistency. Software Engineering Notes **26** (2001) 76–78
4. Hunter, A., B.Nuseibeh: Managing inconsistent specification. ACM Transactions on Software Engineering and Methodology **7** (1998) 335–367
5. A.Hunter: Evaluating the significance of inconsistency. In: Proceedings of the International Joint Conference on AI (IJCAI'03). (2003) 468–473
6. Wiegers, K.E.: Software Requirements,2nd ed. Microsoft Press (2003)
7. 830-1998, I.S.: IEEE Recommended Practice for Software Requirements Specifications. Los Alamitos, CA:IEEE Computer Society Press (1998)
8. A.Hunter: Logical comparison of inconsistent perspectives using scoring functions. Knowledge and Information Systems Journal **6** (2004) 528–543
9. Hunter, A., Konieczny, S.: Approaches to measuring inconsistent information. In: Inconsistency Tolerance,LNCS. Volume 3300. Springer-Verlag (2004) 189–234
10. Hunter, A.: Measuring inconsistency in knowledge via quasi-classical models. In: Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'2002), MIT Press (2002) 68–73
11. D.Dubois, Lang, J., Prade, H.: Possibilistic logic. In: Handbook of logic in artificial intelligence and logic programming. Oxford University Press (1994) 439–531
12. S.Benferhat, D.Dubois, S., H.Prade: Encoding information fusion in possibilistic logic:a general framework for rational syntactic merging. In: Proceedings of ECAI'2000, IOS Press (2000) 3–7
13. S. Konieczny, Lang, J., P.Marquis: Quantifying information and contradiction in propositional logic through test actions. In: Proceedings of IJCAI2003, Morgan Kaufmann (2003) 106–111
14. K.Wiegers: First things first:prioritizing requirements. Software Development **7** (1999) 48–53