Partial Orders and Indifference Relations: Being Purposefully Vague in Case-Based Retrieval

Alex Ferguson Derek Bridge

Department of Computer Science,
University College, Cork
a.ferguson@cs.ucc.ie d.bridge@cs.ucc.ie

Abstract. In this paper, we look at case retrieval systems for product selection. Such systems are interactive. This places demands on the technology: customers must be able to specify their requirements in ways that are meaningful to them; and, the cases that are retrieved must be comprehensible in terms of the customer requirements. To meet these demands, we introduce to case retrieval the notions of similarity metrics with *partially-ordered return types* and of relations that express *indifference between degrees of similarity.* [1]

1 Product Selection

Product selection applications are increasingly being built using case retrieval technology [1]. Products are regarded as the cases. Customers specify preferred values for case attributes, and these values are assembled into a *probe* case. Cases in the case base (products) are compared, using a similarity metric, to the probe and the most similar are retrieved and presented to the customer.

Such systems are *interactive*, and this places the following demands:

The Requirements Criterion: Customers must be able to specify their requirements in ways that are meaningful to them; and

The Results Criterion: Customers must be able to understand, in terms of their requirements, why cases have been included in, or excluded from, the result set.

In this paper, we argue that current case retrieval technology in general does not satisfy either of these criteria particularly well. The problem stems in part from a kind of 'spurious precision' arising from the way that numerically-valued similarity metrics are used in these systems.

Our alternative approach recognises that vagueness and imprecision in customer requirements is a fact of life. They arise from a number of sources, including: the inconvenience to the customer of formulating complete and precise requirements; the customers' lack of knowledge of the product space; and their

[1] This research was funded by Enterprise Ireland (grant number ST/98/024), with the support of Interactive Multimedia Systems, Dublin.

genuine indifference between different portions of the product space. Whatever the source, we feel it is important to model and use this vagueness explicitly.

Our solution is two-fold. First, by use of similarity metrics with partially-ordered return types, we avoid unnecessarily equating quite different notions of similarity. Second, by use of relations that specify indifference between degrees of similarity, we can avoid needlessly distinguishing between different degrees of similarity, when the differences are an artefact of the precision of the similarity metric but are not supported by the customer's own notions of what makes cases similar enough to be meaningfully included together in the same result set.

2 Similarity Metrics

In our framework, similarity metrics, $\sim$, are functions of the following type:

$$\sim :: \alpha \rightarrow \alpha \rightarrow P$$

where P is a partially-ordered set, i.e. $P = (S, \sqsubseteq)$. In words, a similarity metric takes one value of type α and another value of type α and computes their degree of similarity, which will be represented by a value from some set S that has a partial order $\sqsubseteq$ defined on it. The ordering on S is needed so that we can, e.g., determine when one degree of similarity is higher than another.[2]

In most current case retrieval systems, similarity metrics are actually of type:

$$\sim :: \alpha \rightarrow \alpha \rightarrow ([0,1], <)$$

i.e. degrees of similarity are denoted by numbers between 0 and 1 inclusive. This is subsumed by our framework.

It is useful to use the more general form when doing research into similarity. Then, outcomes of the research apply not only to conventional numeric-valued similarity metrics but also to other ways of measuring similarity. For example, Plaza proposes that first-order feature structures can denote degrees of similarity [5]. Plaza's feature-structure-valued functions are subsumed by our framework and so our research applies to his similarity metrics as well as to more conventional similarity metrics.

Our past papers more fully explore the advantages of working in this more general framework, e.g. [4], [2], [3]. In this paper, we will mostly use the more familiar numeric-valued similarity metrics where possible. But we will exploit the power of the more general framework when we come to combine component (attribute-specific) similarity metrics into composite (overall) similarity metrics for whole cases (section 7).

3 Choosing the Result Set: 'best-n'

Consider a case base of products, one of whose attributes is the price of the product. Assume the knowledge engineer believes that prices will not exceed

[2] Note that, in our framework, it is the *result type* that is a partially-ordered set, and not the *cases*.

£1000, $MAX = 1000$. If we take p to be the customer's probe value and c to be the value of a case's price attribute, then one possible similarity metric is the 'less is better' similarity metric from [1], e.g.:

$$c \sim p \hat{=} \begin{cases} 1, & \text{if } c \leq p; \\ 0.8 \times \frac{MAX - c}{MAX - p}, & \text{if } c > p. \end{cases}$$

Suppose the customer decides to retrieve cases based only on their price similarity. (Subsequent sections of the paper will deal with the situation where customers specify preferred values for more than one case attribute.)

The 'less is better' similarity metric, like most numerically-valued metrics, has a high degree of numeric precision. Quite small differences in the computed degrees of similarity may arise. For example, if the customer's probe price is £600, then cases whose prices are £650 and £651 are similar to the probe to degrees of 0.7 and 0.698 respectively. If one of these two cases is in the result set, then the other probably ought to be as well: it is unlikely that such a small numerical distinction would be regarded as decisive by the customer; to exclude only one of them from the result set would probably not reflect the customer's intent. This is well-recognised in the CBR community. But the methods used to overcome this 'spurious precision' merely finesse the problem.

Perhaps most commonly, case retrieval systems mitigate the problem by presenting to the customer a number, n, of the highest-scoring cases. This is convenient but not entirely satisfactory.

Suppose n is 10: the top 10 cases (according to the price similarity metric) will be displayed to the customer. The problem with this approach is that the cases that the customer sees are determined arbitrarily by the choice of n. For example, the top-ranked case and the second-ranked case might have similarities to the probe that are quite close, e.g. 0.7 and 0.698. They are not decisively different cases, so it feels right that both are in the result set. But the cases ranked 10th and 11th might have equally close degrees of similarity, say 0.2 and 0.198. These are not decisively different cases, and yet, for $n = 10$, the 11th-ranked case would not be not included in the result set.

It is also possible that there are large discontinuities in the degrees of similarity to the probe of the cases in the result set. For example, the 1st, 2nd and 3rd-ranked cases might all have very close degrees of similarity, e.g. 0.7, 0.698 and 0.696. But suppose the similarity to the probe of the 4th-ranked case is 0.3. This is quite different from that of the 3rd-ranked case. The requirement to include $n = 10$ cases in the result set dictates that cases such as these, which, from the customer's point of view, are decisively less similar to the probe than the top 3 cases, should nevertheless appear in the result set.

We feel that 'best-n' is an approach that fails to meet the Requirements and Results Criteria. As part of specifying their requirements, customers or knowledge engineers will have to guess a value for n. What they want is an n that will retrieve cases that share a high degree of similarity to the probe. But there is no informed way of choosing this value. Instead, in CBR, we typically use a value that gives a 'screenful' of cases. But this gives the problems described in

the previous paragraph. In interpreting result sets, customers must therefore keep in mind that, because the value for n is arbitrary, the result set may include cases that are decisively different or exclude cases that are not decisively different.

(Of course, this is not a decisive condemnation of 'best-n'. Customers may be comfortable with the idea that the system will always display a screenful of cases; and, the problems with the arbitrariness of what is included and excluded are lessened by making provision for iterative query revision.)

An alternative solution to the problem of 'spurious precision' is to use integer division on the prices before computing their similarities.[3] Integer division 'pre-processes' the case data so that cases with quite similar attribute values are placed into equivalence classes. For example, suppose we use integer division by 10 on the example prices from earlier. After integer division by 10, the two prices £650 and £651 are both 'reduced' to 65, so their degrees of similarity to the probe (originally £600 but also 'reduced' to 60) are both 0.7.

At first glance, this appears to be doing the right kind of thing. Cases that are not decisively different are receiving the same degrees of similarity to the probe. But this is illusory.

Consider the following four prices £649, £650, £651 and £659. Using the integer division approach, the similarities are 0.72, 0.7, 0.7 and 0.7 respectively. Some cases with quite similar prices (e.g. the cases whose prices are £649 and £650) receive different degrees of similarity (0.72 and 0.7) at the same time that cases that differ more in their prices (the cases whose prices are £650 and £659) receive the same degrees of similarity (0.7).

In the next section, we present indifference relations as an alternative to 'best-n' and to the use of integer division. When we define 'indifference' below, we choose not to require transitivity, precisely so that we avoid the problems mentioned in the previous paragraph.

4 Choosing the Result Set: indifference relations

We propose the idea of *indifference relations* an an alternative approach that better achieves the Requirements and Results Criteria. Using indifference relations, we can ensure that the cases in the result set are ones that the customer perceives as related. The $(i+1)$th-ranked case will only appear in the result set if its similarity to the probe is, according to the customer, indistinguishable from the similarity to the probe of the ith-ranked case. On occasion, the result set may be smaller than $n = 10$; on other occasions it may be larger. But, in all cases, inclusion or exclusion will be meaningful and not arbitrary.

Indifference then will be a relation between degrees of similarity. In the price example, it does not compare prices; it compares their similarities. We will denote an indifference relation by $\simeq$. Given two degrees of similarity, s_1 and s_2, $s_1 \simeq s_2$ returns **True** iff s_1 and s_2 are not decisively different degrees of similarity. The customer or knowledge engineer will need to define this relation for each

[3] The criticisms we make of integer division on prices apply equally well to truncating or 'rounding-off' similarity values.

attribute. Their task is to define under what circumstances a customer would be indifferent between two degrees of similarity for an attribute.

The result set that will be displayed to a customer will no longer be the 'best-n'. Suppose the cases that are *most* similar to the probe are similar to the probe to degree m. The result set will include all cases whose similarity to the probe is m or 'close enough' to m. The indifference relation tells you whether a degree of similarity is 'close enough' to m.

The formal definition is written in a way that can handle similarity metrics having any partial-order as their return type:

$$\textit{Retrieve(CB, p, } \sim\textit{, } \simeq\textit{)} \mathrel{\hat{=}} \{c : c \in \textit{CB}, \exists m \in M : c \sim p \simeq m\}$$

$$\text{where } M \mathrel{\hat{=}} \max\{c \sim p : c \in \textit{CB}\}$$

In words, for case base *CB*, probe p, similarity metric $\sim$ and indifference relation $\simeq$, if M is the set of maximal degrees of similarity to p, the cases to retrieve are those whose degrees of similarity are sufficiently close to a value in M (as determined by $\simeq$).

A good example of an indifference relation is ϵ-equality, i.e. we are indifferent between two degrees of similarity, s_1 and s_2, iff $\text{abs}(s_1 - s_2) < \epsilon$. (As we explain in section 5, ϵ-equality is not an equivalence relation. It thereby avoids the problems we encountered with integer division in section 3.)

Suppose there are cases whose prices are £649, £650, £651 and £659. Their similarities to a probe of £600 are 0.702, 0.7, 0.698 and 0.682 respectively. The case whose price is £649 is the most similar to the probe, so this case is in the result set. If we take ϵ to be 0.003, the result set will also include the case whose price is £650. This is in the result set because we are indifferent between the degrees of similarity 0.702 and 0.7 (they are less than 0.003 different). The other cases are not in the result set: their degrees of similarity are not close enough to that of the best case. Of course, if we take ϵ to be 0.005, then the case whose price is £651 will also be in the result set.

We believe that using indifference in this way better satisfies the Requirements and Results Criteria. Specifying an indifference relation is meaningful, and it has a meaningful effect on the result set: non-maximally similar cases are included in the result set only if their similarity falls short of the similarity of a maximally similar case by an amount about which the customer is indifferent.

In fact, this whole scenario can be made even more meaningful by using similarity metrics with more meaningful return types, for example:

$$c \sim p \mathrel{\hat{=}} -\text{abs}(c - p)$$

— the negation of the absolute difference. When the case value equals the probe value, the degree of similarity is at its highest, i.e. zero. The more dissimilar the case value is from the probe, the lower the similarity, as denoted by negative numbers of larger magnitude. The similarity of £650 to £600 is -50, which is higher than the similarity of £651 to £600, which is -51.

In this metric, the return type is not normalised to $[0, 1]$. This exploits a little of the power of our similarity metric framework, which allows any partially-ordered set as the return type. But, furthermore, the degrees of similarity are themselves now more meaningful to customers. They are sums of money (negated so that the distance function becomes a similarity function).

Specifying an indifference relation now becomes more meaningful too. A customer who takes ϵ to be 5 is saying that s/he wants to see the best cases, but s/he would not wish to exclude a case from the result set for the sake of a difference in price of £5. With only a little domain knowledge, meaningful *context-sensitive* choices for ϵ can be made: when looking to purchase a house, ϵ might be £2000; when looking to purchase cars, ϵ might be £400 for one customer or it might be £800 for another customer; when looking at monthly rents, ϵ might be £20 on one occasion and £30 on another occasion. The decision can be made meaningfully by the customer in a customer- and goal-specific way.

The next section makes more precise what we mean by indifference by considering its axiomatisation.

5 Axiomatisation of Indifference

In general, if we are to avoid the kinds of problems we encountered when using integer division in section 3, we do not want indifference relations to construct equivalence classes of similarity values. Indifference relations will generally be weaker than equivalence relations. It is reasonable to require that they be reflexive and symmetric, but they will not in general be transitive.[4] Instead, we define a weaker property that plays a similar role to transitivity, and which we refer to as *convexity*. We define a relation R to be P-convex over any partial order P iff:

$$\forall x, y, z \in P : x \sqsubseteq_P y \sqsubseteq_P z, x\, R\, z \Rightarrow x\, R\, y, y\, R\, z$$

Given three ordered 'points', if the top and bottom points are related by R, then the middle point must also be related to both. We require this property to ensure that there are no 'discontinuities' in the behaviour of indifference relations.

ϵ-equality is reflexive and symmetric. It is not transitive: with $\epsilon = 2$, we are indifferent between values 3 and 4, and between 4 and 5, but not between 3 and 5. It is not, therefore, an equivalence relation. But, it does satisfy convexity: if $x \leq y \leq z$ and if x and z are within ϵ of each other, then x and y are within ϵ of each other, and y and z are within ϵ of each other.

6 Combining Similarity Metrics: weighted averages

For the rest of this paper, we consider the situation where the customer supplies preferred values for several attributes, although, for ease of exposition, we

[4] The question of whether *indifference* is, e.g., symmetric is orthogonal to the question of whether *similarity* is symmetric.

consider the case of two attributes. We need to combine the two *individual* judgements into an *overall* judgement. The two *component* similarity metrics need to be combined into a single *composite* metric.

The conventional approach is to take a (weighted) average of the two (numeric) degrees of similarity. Using an unweighted average is supposed to allow the two component metrics to contribute equally to the composite similarity judgement; using weighted metrics with unequal weights is supposed to allow relative importances of the attributes to be taken into account.

Taking an arithmetic average is an attractively simple solution. But it immediately raises the question of whether this produces a composite metric which reflects the intuitively desired properties (such as allowing component metrics to contribute equally or allowing one to be more important than the other), or indeed whether the composite is meaningful in any strong sense at all. In particular, one might be seen to be 'comparing apples and oranges'; the more differently each metric measures the similarity of its respective attribute, the less clear it is to see whether a particular degree of similarity computed by one metric really equates to a degree of similarity computed by the other metric.

Suppose, for example, that the case base contains hotel descriptions. Our probe price is £60 and our probe comfort-level is '3-stars'. Suppose there is a case in the case base describing a '1-star' hotel offering rooms for £58. Using negated absolute difference, its similarity to the probe on price is -2; its similarity to the probe on comfort is also -2. But few customers would regard these two degrees of similarity as directly comparable, and even fewer would regard the matches on the two attributes to have been successful to the same degree.

Of course, such problems are well-recognised. The usual solution is to perform some sort of normalisation; each component metric is post-processed, so as to conform to some similar pattern. But we are going to claim that this finesses the problem and the real solution is to find out from the customer how different degrees of similarity for different attributes can be 'traded' against each other.

The simplest approach to normalisation is to range-normalise, typically to the interval $[0, 1]$. The return type of the metric is re-scaled so that its minimum possible value is mapped to 0, and its maximum to 1, and the remainder is a linear interpolation between these points. (This is built into the 'less is better' metric given in section 3, for example.) One difficulty is immediate: if no clear bounds exist on the return type of the original metric (due to a lack of such bounds on the case data), then some guesswork is required. Price attributes are a common example of this: knowledge engineers have to pick an arbitrary maximum price (and perhaps even revise it in the light of price inflation!).

Somewhat more subtly, the metric that results from range-normalisation depends very much on the range chosen. Specifically, the magnitudes of the degrees of similarity will depend on the range. A knowledge engineer might feel obliged to use a wide range if 'rogue' values (extremely high or low values) appear in one or other of the case attributes; other knowledge engineers might remove rogue values from consideration when normalising [7]. In the absence of clear bounds, a cautious knowledge engineer who over-estimates a maximum value will ob-

tain a different metric from a less cautious knowledge engineer. In one metric, a narrow normalisation range may have given coarse degrees of similarity; in the other metric, a wider normalisation range may have given fine-grained degrees of similarity; in such circumstances, it is not reasonable to assume that the ways these metrics measure similarity have been made 'equivalent'. This means that there can be just as much of a problem in comparing degrees of similarity for different metrics, even after range-normalisation.

An alternative to range-normalisation, which may take these difficulties into account, is to re-scale according to statistical properties of a metric, if these are known or can at least be approximated. This approach will typically convert a function with a given mean and standard deviation to a function with a predetermined mean and standard deviation. This is more 'stable' than range-normalisation, but to deploy it also requires somewhat better-quality information than does range-normalisation. But, for product selection applications in particular, there remains a fundamental problem, as discussed in the next paragraph.

An assumption underlying these different methods of normalisation is that 'equivalence' of degrees of similarity from different similarity metrics is at bottom a matter of pro-rating one consideration against another. This is seen as a technical problem: how to determine what this pro-rating factor might actually be (e.g. based on range, based on range but ignoring rogue values, based on statistical measures, etc.). But there is a more fundamental problem: in product selection applications in particular, finding this pro-rating is not just a mathematical exercise; it is also a cognitive exercise. The assumption that a technical, mathematical 'fix' will tell us how a customer would be prepared to 'trade-off' differences in similarity values in different metrics is at least questionable. In product selection applications, where we are concerned with subjective user preferences, the assumption is likely to be wrong.

Suppose we could elicit some notion of metric sensitivity: how strongly the customer feels about a change in one metric's similarity value, as compared with a change in another metric's similarity value. If we could determine this, we could normalise by the same factor. Then, normalisation would not be treating the metrics simply as mathematical objects, but would be taking into account customer perceptions. But, such an approach could involve considerable up-front elicitation groundwork. (And, this burden is made all the greater because, in general, the normalisation factor would be customer- and goal-specific.) For these reasons, in CBR, we tend to take the mathematical approach to normalisation, rather than the cognitive approach described in this paragraph.

Finding the right normalisation is only part of the problem. Averaging brings problems too. A first problem is that combining two numbers into a single number loses information that the customer might have found useful. If the customer could see the judgements of the component metrics, s/he could make more informed product selection decisions or query revision decisions.

A second, related problem is that averaging cannot be *agnostic* in the way it combines even conflicting judgements: the degrees of similarity it obtains are always totally-ordered. If case 1 is more similar to the probe on one attribute

than is case 2 but, at the same time, it is less similar to the probe on the other attribute, then *in the absence of any other information from the customer* (and in the light of the problems of 'equating' degrees of similarity in normalisation), agnosticism about the ranking of these two cases would be more appropriate than forcing an ordering onto them.

Finally, if we accept that component metrics exhibit 'spurious precision', then a composite metric, formed using averaging, will also exhibit 'spurious precision': the precision in the component metrics may not reflect customer perceptions, so the precision of the composite metric is also unjustified (especially if it is even more precise than that of the components).

The weights used in averaging are another problem. Contrary to popular belief, weights are hard to interpret. Setting the weight of one similarity metric to 1 and another to 2 does not, in general, give the second attribute twice the weight of the first. One reason for this is the failure to start from 'a level playing field'. If normalisation has not adequately equated similarity values from the different similarity metrics, then one or other of the metrics may count for less at the outset, and weighting may simply increase or decrease this initial imbalance.

On top of this, weights have no real meaning in terms of the original similarity metrics. Customers who are asked to choose weights (or indicate attribute importances in some way) can only do so in an uninformed way. They have no clear idea of the effect their weights will have. For example, if a customer is viewing a result set and decides to re-issue the original query but with new weights, the customer has no way of knowing in advance whether the new result set will be identical to the original, a re-ranked version of the original, completely different from the original, or partly the same as the original.

In conclusion, conventional approaches to combining similarity metrics compound at least four levels of arbitrariness and 'spurious precision'. First, the original metrics are often spuriously precise (as argued earlier in this paper). Second, the normalisation is often arbitrary and aims to satisfy mathematical objectives instead of cognitive ones. Third, the use of averaging compounds the precision of the original metrics, loses information and cannot be agnostic. Fourth, weights cannot, in general, be used to achieve definite effects.

In the next section, we present a different way of combining similarity metrics. By using return types that are partially-ordered sets of pairs, it requires no normalisation, and it does not lose any information. And, by using indifference relations, it allows customers to specify trade-offs between degrees of similarity.

7 Combining Similarity Metrics: generalised prioritisation

Our similarity metric framework allows similarity metrics to have as their return types any partially-ordered sets. In particular, we can use *pairs of values* to denote degrees of similarity, i.e. the return type of a similarity metric could just as well be a partially-ordered set of pairs as it could be a set of numbers or Plaza's first-order feature structures.

This proves especially useful when combining component similarity metrics into composite similarity metrics. For example, suppose cases contain a price attribute and a size attribute. Suppose the probe specifies a price of £600 and a size of 100. Consider case 1 that has a price of £550 and a size of 80. Using negated absolute difference, the similarity of case 1 to the probe on price is -50 and on size is -20. By the conventional approach, to form the composite degree of similarity, the two similarity values would be normalised, and the values would be averaged. But, in our approach, where we exploit the ability to have any partially-ordered set as the return type of a similarity metric, the composite metric returns pairs of values. In this approach, the similarity of case 1 to the probe is $\langle -50, -20 \rangle$: the pair *is* the degree of similarity. There is no normalisation; there is no combining into a single numeric value; so there is no loss of information.[5] If case 2 has a price of £640 and a size of 130, its similarity to the probe is the pair $\langle -40, -30 \rangle$.

Of course, this is not enough. Had we averaged the similarity values we would be able to see which of case 1 and case 2 was the more similar to the probe. If we use pairs to denote degrees of similarity, we similarly need to be able to compare the pairs. This is why the return type of a similarity metric has an ordering. This ordering on the pairs can be any ordering you wish to define, but most usually it will be defined from the orderings of the original component similarity metrics.

In the next few subsections, we show three general ways in which customers can form this ordering. Each is meaningful. We will take the return type of the first composite similarity metric to be the partially-ordered set $(A, \sqsubseteq_A)$ and we will take the return type of the other metric to be $(B, \sqsubseteq_B)$.

7.1 Product

Suppose the customer wishes the first metric to contribute neither more nor less than the second. To achieve this, the customer would define an ordering on the pairs using an operator that we call *product*:

$$\langle x_1, x_2 \rangle \sqsubseteq_{A \times B} \langle y_1, y_2 \rangle \mathrel{\hat{=}} x_1 \sqsubseteq_A y_1 \wedge x_2 \sqsubseteq_B y_2$$

i.e. for one pair to be lower in the ordering than another, both its components must be lower in their respective orderings.

It is possible, of course, for a case to be better on one component and worse on the other. The definition of product says that these pairs are incomparable, and they are thus, in some sense, 'as good as' each other. In the absence of any information about how we might 'trade-off' A against B, exploiting the potential in a partial order for incomparability seems the right thing to do. The numerical averaging approach can never be agnostic in this way.

[5] Our earlier papers have pointed out that another advantage of this approach is that we can combine similarity metrics that have quite different return types (see, e.g., [4]). For example, a similarity metric whose return type is numeric could be combined with one of Plaza's feature-structure-valued metrics *without inter-conversion*. This is something the averaging approach cannot do.

7.2 Strict Prioritisation

Consider now the case of a customer who regards one of the attributes as primary and the other as secondary. Specifically, the secondary requirement is to be met as much as possible but only after the primary requirement has been met as much as possible [6]. (The case where the primary attribute is more important but not *absolutely* more important will be dealt with in the next subsection.)

All that the customer needs here is a new ordering on the similarity pairs. We call the operator that forms this ordering *strict prioritisation*:

$$\langle x_1, x_2 \rangle \sqsubseteq_{A \gg B} \langle y_1, y_2 \rangle \mathrel{\hat{=}} x_1 \sqsubset_A y_1 \vee (x_1 = y_1 \wedge x_2 \sqsubseteq_B y_2)$$

The overall ordering is based on the primary ordering ($\sqsubseteq_A$) but the secondary ordering ($\sqsubseteq_B$) may resolve ties in the first ordering.

Weighted averages can make one attribute more important than another, but they cannot so easily make one attribute *absolutely* more important than another. Consider, for example, combining three metrics, where the first is absolutely more important than the second and the second is absolutely more important than the third. Choosing a weight for the second metric is hard: it has to be large enough to guarantee that the second metric will be absolutely more important than the third metric, but small enough to guarantee that it will be absolutely less important than the first metric. Particular weighting schemes can be crafted for particular metrics. But there is no *general* approach using weighting, and this is the value of strict prioritisation.

7.3 Generalised Prioritisation

Customers have another way of combining partial-orders, which we call *generalised prioritisation*:

$$\langle x_1, x_2 \rangle \sqsubseteq_{A \gg_\simeq B} \langle y_1, y_1 \rangle \mathrel{\hat{=}} x_1 \sqsubseteq_A y_1 \wedge (x_1 \simeq_A y_1 \Rightarrow x_2 \sqsubseteq_B y_2)$$

As you can see, this definition offers another use for indifference relations, $\simeq$. It gives us an ordering which, to some extent, favours its first argument, but not necessarily to the extent of strict prioritisation. We order the pairs according to the first value in the pair, unless we are indifferent between the first values, in which case we take both components of the pair into account.

This is the second distinct use of indifference in this paper. As before, we claim that it satisfies the Requirements and Results Criteria quite well. We hope that customers will find indifference meaningful during specification of requirements. And, they should be able to interpret the results they see in terms of their indifference relation: there can be worse matches on the first metric (worse, but only up to the customer's level of indifference) if they are compensated for by better matches on the second metric.

In passing, we should point out that generalised prioritisation has the nice property that it has product and strict prioritisation as special cases. If we prioritise order A over order B but we define $x \simeq_A y = \mathbf{True}$, then we obtain

an ordering identical to $\sqsubseteq_{A\times B}$, the product of the two orders. If we prioritise order A over order B but we use equality for $\simeq_A$, then, after simplification, the definition reduces to our previous definition for strict prioritisation, $\sqsubseteq_{A\gg B}$.

8 Conclusion

We have argued that conventional case retrieval technology exhibits a level of unjustified precision which can complicate specification of customer requirements and interpretation of retrieval results in terms of those requirements.

We have proposed an approach that uses meaningful, partially-ordered return types for similarity metrics (especially for composite metrics) and a notion of 'indifference' between degrees of similarity. By accommodating partial-orders, customers are not obliged to use over-specified metrics. And by using indifference relations, we give explicit control over the precision of metric return types.

We have built a case retrieval system that works in the way described in this paper. To the user, the system looks much like conventional case-retrieval systems. The knowledge engineer will have supplied the individual similarity metrics. A choice of combining operators (section 7) must be made and definitions of indifference must be given (which can sometimes be as simple as specifying a value for ϵ). The results of a query again look much as they do in other systems. The difference is that the result set need not be of some arbitrary predetermined size (although, since our framework subsumes conventional approaches, customers can issue 'best-n' queries if they really wish to). The system offers a rich set of options for query revision [3].

References

1. Bergmann, R., Breen, S., Göker, M., Manago, M. & Wess, S.: *Developing Industrial CBR Applications*, LNAI-1612, Springer, 1999
2. Bridge, D.G.: Defining and Combining Symmetric and Asymmetric Similarity Measures, in B.Smyth & P.Cunningham (eds.), *Advances in CBR (Procs. of 4th European Workshop on CBR)*, LNAI-1488, pp.52-63, Springer, 1998
3. Ferguson, A. & Bridge, D.: Options for Query Revision when Interacting with Case Retrieval Systems, in I.Watson (ed.), *Procs. of the Fourth UK CBR Workshop*, University of Salford, 1999
4. Osborne, H. & Bridge, D.: Similarity Metrics: A Formal Unification of Cardinal and Non-Cardinal Similarity Measures, in D.B.Leake and E.Plaza (eds.), *CBR Research and Development (Procs. of Second International Conference on CBR)*, LNAI-1266, pp.235-244, Springer, 1997
5. Plaza, E.: Cases as terms: A feature term approach to the structured representation of cases, in M.Veloso & A.Aamodt (eds.), *CBR Research and Development (Procs. of First International Conference on CBR)*, LNAI-1010, pp.265-276, Springer, 1995
6. Vollrath, I.: Handling Vague and Qualitative Criteria in Case-Based Reasoning Applications, in the on-line pre-proceedings of the Eighth German Workshop on CBR, 2000 (`http://wwwagr.informatik.uni-kl.de/~gwcbr2k/program.html`)
7. Wilson, D.R. & Martinez, T.R.: Improved Heterogeneous Distance Functions, *Journal of Artificial Intelligence Research*, vol.6, pp.1-34, 1997