*Artificial Intelligence Journal, Volume 150, November 2003, pp. 145-181*

# Extensionally Defining Principles and Cases in Ethics: an AI Model

**Bruce M. McLaren**

e-Supply Chain Management Laboratory
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891

## Abstract

Principles are abstract rules intended to guide decision-makers in making normative judgments in domains like the law, politics, and ethics. It is difficult, however, if not impossible to define principles in an intensional manner so that they may be applied deductively. The problem is the gap between the abstract, open-textured principles and concrete facts. On the other hand, when expert decision-makers rationalize their conclusions in specific cases, they often link principles to the specific facts of the cases. In effect, these expert-defined associations between principles and facts provide *extensional* definitions of the principles. The experts *operationalize* the abstract principles by linking them to the facts.

This paper discusses research in which the following hypothesis was empirically tested: extensionally defined principles, as well as cited past cases, can help in predicting the principles and cases that might be relevant in the analysis of new cases. To investigate this phenomenon computationally, a large set of professional ethics cases was analyzed and a computational model called SIROCCO, a system for retrieving principles and past cases, was constructed. Empirical evidence is presented that the operationalization information contained in extensionally defined principles can be leveraged to predict the principles and past cases that are relevant to new problem situations. This is shown through an ablation experiment, comparing SIROCCO to a version of itself that does not employ operationalization information. Further, it is shown that SIROCCO's extensionally defined principles and case citations help it to outperform a full-text retrieval program that does not employ such information.

# 1 Introduction

In domains such as ethics and the law, principles, norms, or laws govern human behavior and inform reasoning and decision-making. While the consequences of violating abstract rules such as these can be far-reaching and severe, the rules themselves are often quite difficult to apply. Typically, abstract rules contain open-textured terms [Twining and Miers, 1976; Gardner, 1987] that cover a wide range of specific fact situations. The rules are subject to interpretation and may have different meanings in different contexts. Frequently, the terms are not defined precisely. There is a gap between the abstract rules and the concrete facts to which they apply. Often, there are no authoritative or readily available intermediate rules with which to "bridge the gap," that is, no rules of finer granularity that elaborate the abstract rules' meaning or explain how they apply in concrete circumstances. As if that were not enough, frequently more than one abstract rule applies to a fact situation with conflicting advice.

Consider, for example, the following engineering ethics principles, taken from the National Society of Professional Engineers reference guide [NSPE, 1996]:

> NSPE Code I.1: "Engineers, in the fulfillment of their professional duties, shall hold paramount the safety, health, and welfare of the public in the performance of their professional duties."
>
> NSPE Code III.2.b: "Engineers shall not complete, sign or seal plans and/or specifications that are not of a design safe to the public health and welfare and in conformity with accepted engineering standards."
>
> NSPE Code II.1.c: "Engineers shall not reveal facts, data, or information obtained in a professional capacity without the prior consent of the client or employer except as authorized or required by law or this Code."

While most people would agree that abstract principles such as these are reasonable and appropriate, it is difficult to apply them in real-world situations [Jonsen and Toulmin, 1988]. Since the principles contain open-textured terms and phrases (e.g., how does one precisely define "fulfillment of professional duties," "design safe to the public health," or "prior consent"?), it is not possible for experts to define intermediate-level rules to cover all possible conditions to which the principles apply. While expert ethicists' interpretations and applications of moral codes may be compelling, they are not necessarily authoritative. Often even the principles' recommended actions are abstract (e.g., what exactly does it mean to "hold paramount" the safety, health, and welfare of the public?). Moreover, one can easily imagine fact situations pitting an engineer's obligation to public safety against his obligation to maintain his client's confidences. More than one of these

principles may appear to apply equally well to the fact situation yet lead to conflicting conclusions and recommendations.

For all of the reasons above – open texture, lack of authoritative intermediate rules, and conflicts among abstract rules – problem solving in ethics and law can be characterized as ill-defined. The difficulty of using deductive logic to address problems in such ill-defined domains has long been recognized [Toulmin, 1958; Berman and Hafner, 1986; Gardner, 1987]. It appears to be infeasible to define the abstract rules intensionally in precise and complete enough terms for a deductive problem solver to apply them correctly to myriad fact situations or to resolve the resulting conflicts[1].

Nevertheless, ethical and legal decision makers do apply such principles more or less systematically. Often, they record why certain principles apply to particular fact situations and why some principles trump other principles in those fact situations.

The work described in this paper was based on the hypothesis that, as ethicists record their explanations of how and why they applied and reconciled the principles in resolving specific fact situations, they *extensionally* define the principles. In other words, the expert's explanations effectively *operationalize* the principles [Mostow, 1983]. Thus, it appeared possible that a computational model could leverage these operationalizations, if not actually to resolve new ethical dilemmas, then at least to retrieve and predict principles and past cases that are relevant to the analysis of new problems.

In order to test this hypothesis, a computational model called SIROCCO (**S**ystem for **I**ntelligent **R**etrieval of **O**perationalized **C**ases and **CO**des) was constructed. SIROCCO is a program that represents some of the extensional connections between abstract rules and scenarios made by decision-makers as they apply operationalization techniques. The program uses these connections to retrieve principles and past cases that are relevant to new fact situations. SIROCCO was empirically tested by comparing its performance to a version of itself without operationalization information and also to a comparable full-text retrieval program. This paper describes how SIROCCO operates, its experimental evaluation, and the research contributions of the work.

## 1.1 A Guide to the Paper

The paper is organized as follows.

Section 2 discusses the specific domain of engineering ethics, particularly the principles in the NSPE Code of Ethics and the cases decided by its Board of Ethical Review (BER). Section 2 also

[1] By contrast, work in deontic logic does attempt to define abstract rules in an intensional manner. Deontic logic and its potential application to the ethics problem domain are discussed in Section 7, "Comparison to Related Work."

introduces the nine operationalization techniques that were uncovered during analysis of the domain. Finally, it is suggested that the operationalization of abstract principles plays a role in well-defined domains, such as classroom physics.

Section 3 presents an overview of SIROCCO, including a description of its ontology, its case base, and how it uses the operationalizations to retrieve relevant information. An extended example based on an actual NSPE BER case shows how the facts of a case (Section 3.1), abstractions of the facts and principles (Section 3.2), and the board's analysis (Section 3.3) are represented so that SIROCCO can reuse them. A summary of SIROCCO's case base is provided in Section 3.4 and a detailed description of SIROCCO's two-stage retrieval algorithm, using the example case as a basis, is provided in Section 3.5. SIROCCO's output for the example case is also shown in Section 3.5.

Section 4 describes the SIROCCO experiments. The purpose of the experiments was to test the hypothesis that operationalized principles and past cases can help predict the principles and cases that are relevant in analyzing new cases. Among a series of experiments, SIROCCO is compared to a full-text retrieval system and to a version of itself lacking the operationalization information.

Section 5 presents the quantitative results of the experiments, including the fact that SIROCCO significantly outperforms both the full-text retrieval program and the ablated version of itself.

Section 6 analyzes and discusses the experimental results, most importantly the confirmation of the hypothesis. This section describes how the experimental results clearly demonstrate the importance of operationalizations for predicting relevance.

Section 7 discusses related work. SIROCCO is compared to other interpretive CBR programs, in particular those from the field of AI and Law, such as GREBE, CATO, HYPO, and BankXX. A comparison of the domain of professional ethics and the law is also provided in this section.

Section 8, the concluding section, discusses the two major contributions of the work to AI and Cognitive Science. In particular, (1) the concept and use of operationalization of abstract principles in an ill-defined domain and (2) the example the SIROCCO project provides of how an AI model and methods can be used to perform empirical research in an ill-defined domain. Section 8 also discusses subsequent work imbuing SIROCCO with the capability to "know what it knows" and briefly discusses plans for using the program as a retrieval component in a tutoring system for engineering students.

# 2 Operationalization in the Ethics Domain

This research involved a systematic study of the published opinions of the NSPE and its code of ethics [NSPE, 1958-1998]. The NSPE Board of Ethical Review, composed of five to seven

professional engineers annually, has written extensive explanations of how and why codes apply or do not apply to more than 400 fact situations. The NSPE's code of ethics, consisting of 75 principles including the three cited above, provides engineers with guidance on a wide range of ethical issues such as protecting public safety, employment duties, fairness in advertising, conflicts of interest, and confidentiality. The NSPE BER database provides a valuable record of how engineering ethics experts believe engineering ethics codes apply to practical situations. The full set of cases between 1958 and 1998, as well as the principles that applied to those cases, may be found at the following web site: www.pitt.edu/~bmclaren/ethics.

Nine heuristic techniques were repeatedly used by the BER to justify their conclusions. Each of the techniques involved the citation and interpretation of either NSPE code provisions or past cases and effectively operationalized those codes and cases. The specific techniques uncovered during the analysis are shown in Figure 1.

1. *Principle Instantiation*: Instantiating a principle by linking it to clusters of questioned and critical facts.
2. *Fact Hypotheses*: Hypothesizing facts that affect how a principle applies.
3. *Principle Revision*: Revising a principle over time in light of new cases or changes in culture.
4. *Conflicting Principles Resolution*: Resolving conflicting principles in a specific case.
5. *Principle Grouping*: Grouping principles in a specific case to bolster an argument.
6. *Case Instantiation*: Instantiating a case as a precedent by linking it to clusters of questioned and critical facts, and by analogizing or distinguishing it.
7. *Principle Elaboration*: Applying, defining or elaborating issues and principles from past cases.
8. *Case Grouping*: Grouping cases to bolster an argument.
9. *Operationalization Reuse*: Reusing a specific application of any of the above techniques from previous analyses.

**Figure 1**: Operationalization Techniques used by the NSPE BER

Although the board applied the techniques in an implicit manner, textual clues indicated where the techniques had been applied. I catalogued specific examples of how the board appeared to apply each technique [McLaren, 1999, Chapter 2].

The analysis confirmed that the operationalization techniques were identifiable and repeated across a large segment of the NSPE cases, and that the techniques in Figure 1 were used to resolve the dilemmas and/or justify conclusions[2]. For example, *Principle Instantiation* was used to cite a

[2] The analysis proceeded in three stages. First, for approximately 50 of the foundational cases, I highlighted discussion text in the NSPE BER analyses that either explicitly or implicitly cited codes, past cases, or important issues and then indicated the purpose of each citation (e.g., "analogy to a past case," "interpreting some terms of the code," "deciding which code takes precedence in the context of this case," etc.). Second, I reread all of the highlighted discussion text to determine whether a canonical set of techniques could be identified. At this stage the operationalization techniques shown in Figure 1 were cataloged. Third, to verify the general applicability of the techniques, a substantial number of additional foundational cases (over 300) were read and less formally analyzed.

principle and then connect that principle to selected facts of a case with the purpose of either bolstering or rhetorically challenging a conclusion. The *Case Instantiation* technique was employed in an analogous fashion; it was used to cite a past case and draw an analogy to that case (through selected facts of the current case) for the purpose of supporting or contradicting a conclusion. *Principle* and *Case Grouping* are techniques in which the board cited *multiple* principles or cases in support of a conclusion. Finally, *Operationalization Reuse* was applied when the board reaches back to a previous analysis and reuses some of its reasoning (specifically, the reasoning associated with one of the eight other techniques) in the analysis of a new case.

The primary goal in building SIROCCO was to test whether it could use a core subset of the operationalization techniques (specifically, the five techniques discussed in the previous paragraph, i.e., *Principle Instantiation*, *Case Instantiation*, *Principle Grouping*, *Case Grouping*, and *Operationalization Reuse*) to make predictions of the principles and past cases that are relevant in the analysis of new cases. In particular, the goal was to see whether the extensional definitions of codes and cases embodied in the NSPE BER case analyses could help SIROCCO in making its predictions. Secondarily, the goal was to test whether the temporal knowledge provided in the chronological narratives would contribute to the program's accuracy.

It is interesting to note that while operationalization appears to be important in ill-defined domains like engineering ethics, it also seems to play a role in well-defined domains such as classroom physics or geometry problem-solving. In fact, Mostow introduced operationalization [1983] in the comparatively well-defined domain of card-playing.

Physics problem-solving is well-defined (at least at the undergraduate level) in that physics problems have definitive answers that can be justified by applying abstract principles whose terms are clearly defined. Nevertheless, problem-solvers in classroom physics need to learn how to operationalize the abstract principles in solving problems. For instance, to solve simple mechanics problems with Newton's Second Law ($F = ma$), a student still needs to learn procedures to figure out which entity should be considered "the body", select a reference frame, find all of the forces acting on the body, and find the acceleration [Resnick and Halliday, 1967, p. 96].

Learning to operationalize a principle requires students to examine many examples; the simple statement of the principle $F = ma$ does not disclose the procedures for operationalizing it. The examples are organized by situation type, for instance, three strings tied together with a knot, with a block attached to one string. The examples instantiate the principle by illustrating the procedures in a specific context. For instance, in the 3-string context, students learn to identify the knot as the body (as opposed to the block). They learn that $F$ in the equation refers to the total force, requiring

all of the individual forces on the knot to be identified and summed. Such procedures, learned by example, effectively operationalize the principle [Pinkus *et al.*, 1997].

Rissland elucidated how processes of constructing examples contributed to understanding mathematical concepts by operationalizing them [1978].

# 3 An Overview of SIROCCO

After completing the domain analysis, I built SIROCCO to computationally model and apply the nine operationalization techniques. Using the techniques as a guide, particularly those that conceptually link case facts to codes or cases in an extensional manner (i.e., *Principle Instantiation* and *Case Instantiation*), SIROCCO retrieves and predicts codes and past cases that may be relevant in the analysis of a new case. As described more fully in [McLaren, 1999], SIROCCO employs an engineering ethics ontology, including a language to describe engineering scenarios as chronological narratives of events. The ontology also includes components used to represent the board's arguments for and against its conclusions. A web-based case acquisition tool assists human case-enterers in the translation of cases into the SIROCCO ontology. A total of 184 cases have been translated into the program's foundational case base, primarily by a group of independent case-enterers and a few by me. An additional 58 trial cases were represented exclusively by independent case-enterers in order to perform the experiments described in this paper.

SIROCCO is an interpretive case-based reasoning (CBR) program [Kolodner, 1993, pgs. 86-92]. Interpretive CBR is a sub-field of case-based reasoning in which complex, ill-structured, and highly linguistic fact situations are evaluated in the context of previous experience. This work is also closely related to AI and Law research [Ashley, 1990; Rissland and Skalak, 1991; Branting, 1991, 2000; Rissland *et al.*, 1996; Aleven, 1997], but it pioneers research in the domain of engineering ethics.

SIROCCO is distinguished from previous AI and Law systems by its detailed, narrative case representation, including temporal relations between facts,[3] and an extensional model of abstract principles and cited cases. It can retrieve cases over a wider range of factual scenarios than the earlier AI and Law programs, but unlike those programs it does not make arguments for or against a conclusion. Rather, it provides suggestions that can help a human construct a reasoned argument. Other distinctions between this work and AI and Law work are discussed in Section 7, "Comparison to Related Work."

[3] Temporal relations have been modeled in CBR as a strict linear sequence of events [Bonissone and Ayub, 1992], but the cases were not represented as narratives and events could not be treated as overlapping or encompassing other events, as in SIROCCO.

SIROCCO operates by (1) accepting a target case expressed in the Ethics Transcription Language (ETL), (2) searching for relevant information in a case base of source cases expressed in the Extended Ethics Transcription Language (EETL), and (3) producing suggested code provisions and past cases, as well as other suggestions. ETL is the constrained, narrative language used to represent target or input cases. EETL is an extension of ETL that supports representation of the argument made by the board for a case and is the language of the source cases. EETL is also the language used to represent the operationalization techniques.

## 3.1 Representing Target Cases: The Ethics Transcription Language

The facts of a sample target case are shown in Figure 2. In this case, Engineer A is confronted with conflicting obligations. He has an obligation to protect the public, but he also has an obligation of confidentiality to his client. The facts are as follows. Engineer A is hired to analyze the structural integrity of an inhabited apartment building. Engineer A's client, the owner of the building, instructs the engineer that his findings are to be held in confidence. The engineer does not find anything structurally wrong with the building, but the client confides that there are problems with the electrical and mechanical systems of the building. Although Engineer A knows these problems could be safety hazards, he does not contact the authorities. Rather, he just mentions these problems in the final report he provides only to the owner. The question raised by this case is whether it was ethical for Engineer A not to report the safety violations to the proper authorities.

***Facts***: Engineer A is retained to investigate the structural integrity of a 60-year old occupied apartment building that his client is planning to sell. Under the terms of the agreement with the client, the structural report written by Engineer A is to remain confidential. In addition, the client makes clear to Engineer A that the building is being sold "as is" and he is not planning to take any remedial action to repair or renovate any system within the building prior to its sale.

Engineer A performs several structural tests on the building and determines that the building is structurally sound. However, during the course of providing services, the client confides in Engineer A and informs him that the building contains deficiencies in the electrical and mechanical systems that violate applicable codes and standards. While Engineer A is not an electrical nor mechanical engineer, he does realize those deficiencies could cause injury to the occupants of the building and so informs the client.

In his report, Engineer A makes a brief mention of his conversation with the client concerning the deficiencies; however, in view of the terms of the agreement, Engineer A does not report the safety violations to any third party.

***Question***: Was it ethical for Engineer A not to report the safety violations to the appropriate public authorities?

**Figure 2**: Sample Target Case (Case 89-7-1)

The ETL representation of Case 89-7-1 is shown in Figure 3. In designing ETL, I chose to represent ethics cases as chronological narratives of events. The intent was to provide a

representation that allowed the modeling of actual engineering events and actions but was still limited enough for SIROCCO to reason about. The tack taken here is similar to that of SWALE [Leake, 1991] and GREBE [Branting, 1991, 2000], except that I focused strictly on representing the facts and not the semantic explanation of the conclusions given those facts. Limiting the complexity of the language was instrumental in allowing independent third parties to represent – and for SIROCCO to process – a wider range of topics than either SWALE or GREBE.

| | |
|---|---|
| 1. Client X **<hires the services of>** Engineer A. | Pre-existing fact |
| 2. Apartment House P **<may be hazardous to safety>.** | Pre-existing fact |
| 3. Client X **<instructs>** Engineer A **<to>** (Engineer A **<withholds information from>** Any Third Parties **<regarding>** Structural Report R). | Occurs during 1, 2 |
| 4. Engineer A **<inspects>** Apartment House P. | Occurs during 1, 2,<br>After the conclusion of 3 |
| 5. Engineer A **<discovers that>** (Apartment House P **<fails standards and may be hazardous to safety.>**) | Occurs during 4 |
| 6. Engineer A **<knows>** (Government Authority **<should be informed about the hazard or potential hazard>**). | Immediately after the conclusion of 5 |
| 7. Engineer A **<informs>** Client X **<that>** (Apartment House P **<fails standards and may be hazardous to safety.>**) | Immediately after the conclusion of 5 |
| 8. Engineer A **<writes paper/article>** Structural Report R **<about>** Apartment House P. | After the conclusion of 7 |
| 9. Engineer A **<provides limited information to>** Client X **<regarding>** (Engineer A **<discovers that>** (Apartment House P **<fails standards and may be hazardous to safety.>**)) | Occurs as part of 8 |
| 10. Engineer A **<does not inform>** Any Third Parties **<that>** (Apartment House P **<fails standards and may be hazardous to safety.>**) *[Questioned fact]* | After the start of 8 |

**Figure 3**: ETL Representation of Case 89-7-1

As enforced by the grammar shown in Figure 4, ETL represents the actions and events of an engineering scenario as a *Fact Chronology*, which is an ordered list of *Facts* (i.e., individual sentences). Each Fact consists of: (1) *Actors and Objects*, instances of general actors and objects that appear in the scenario; (2) a *Fact Primitive,* the action or event in which the actors and objects participated; and (3) *Time Qualifiers*, temporal relations that specify how the Fact relates to other Facts in time. For instance, in the case representation of Case 89-7-1 (Figure 3), the fourth Fact of the Fact Chronology contains the Actor "Engineer A," the Object "Apartment House P," the Fact Primitive "inspects," and the Time Qualifiers "Occurs during 1, 2; After the conclusion of 3." ETL currently supports 70 Actor and Object Types, 190 Fact Primitives, and 10 Time Qualifiers in the

definition of ETL. The entire set of Actors, Objects, Fact Primitives, and Time Qualifiers in SIROCCO is provided in [McLaren, 1999].

| | |
|---|---|
| <Fact-Chronology> := | <Fact> [ <Fact> …] |
| <Fact> := | <Fact-#> <Fact-Phrase> [ (Questioned Fact <X>) ]<br><Time-Qualifier> [,<Time-Qualifier>, ... ] |
| <Fact-Phrase> := | <Fact-Primitive> [<Fact-Modifier>] <Actor-Or-Object><br>[<Actor-Or-Object> \| (<Fact-Phrase>) ]<br>[<Actor-Or-Object> \| (<Fact-Phrase>) ] |
| <Fact-#> := | <Positive-Integer> |
| <Fact-Primitive> := | *An instance of a Fact-Primitive* |
| <Actor-Or-Object> := | *An instance of an Actor or an Object* |
| < Fact-Modifier> := | partially \| substantially \| limited \| extensive |
| <Time-Qualifier> := | Pre-existing fact \|<br>After the start of <Fact-#> [, <Fact-#>, ... ] \|<br>Starts at the same time as <Fact-#> [, <Fact-#>, ... ] \|<br><Time-Period> after the start of <Fact-#> [, <Fact-#>, ... ] \|<br>After the conclusion of <Fact-#> [, <Fact-#>, ... ] \|<br>Immediately after the conclusion of <Fact-#> [, <Fact-#>, ... ] \|<br><Time-Period> after the conclusion of <Fact-#> [, <Fact-#>, ... ] \|<br>Ends <Fact-#> [, <Fact-#>, ... ] \|<br>Occurs during <Fact-#> [, <Fact-#>, ... ] \|<br>Occurs as part of <Fact-#> [, <Fact-#>, ... ] \|<br>Occurs concurrently with <Fact-#> [, <Fact-#>, ... ] \| |
| <Time -Period> := | <Y> Days \| <Y> Weeks \| <Y> Months \| <Y> Years |
| <X> := | Empty \| <Positive-Integer> |
| <Y> := | Many \| Several \| <Positive-Integer> |
| <Positive-Integer> := | 1 ... N |

**Key:** | = Alternative; [ ] = Optional; < > = Grammar element

Regular font indicates literal placement (e.g., "Pre-existing fact")

Italicized font indicates a general description of a terminal item not shown in this figure

**Figure 4**: The Grammar for the Ethics Transcription Language (ETL)

At least one Fact in every chronology is the *Questioned Fact*; this is the action or event corresponding to an ethical question raised in the scenario. In Figure 3, Fact 10 is the Questioned Fact: it describes Engineer A's failure to alert the authorities about his findings. If more than one question is raised in the context of the same Fact Chronology, a typical situation, the Questioned Facts are enumerated (e.g., "Questioned Fact 1," "Questioned Fact 2," etc.) and a separate *Case* is created for each question.

The Time Qualifiers provide a means of relating facts temporally. A working set of these qualifiers was identified during the NSPE case analyses. Allen's temporal relations [1983] were then mapped to the qualifiers so that they could be processed in a principled manner by SIROCCO. Each Time Qualifier is a disjunctive composition of the Allen relations [see McLaren, 1999, Figure

3-8]. SIROCCO uses TIMELOGIC [Koomen, 1989], a time propagation program, to infer temporal relations not provided by the case enterer.

To input a target case, a case enterer must transcribe the textual facts of a case (Figure 2) into a Fact Chronology (Figure 3). To support this process a web-based case acquisition tool was developed (www.pitt.edu/~bmclaren/ethics). The web site helps the case enterer by providing a transcription template, transcription instructions, an example set of almost 50 transcribed cases, and a complete description of the ontology. The web site is highly cross-linked and cross-referenced to further ease the task of case entry.

The goal is to support the case enterer in representing case events as accurately as possible, given the limited set of Fact Primitives, Actors, and Objects. Throughout the development of SIROCCO and the transcription of the foundational cases, the ETL vocabulary was gradually supplemented. For instance, the case enterers identified important missing Fact Primitives and Actor or Object Types. However, after the transcription of the 184 foundational cases, the vocabulary size appeared to have leveled off, at least for the range of engineering ethics cases that were part of the NSPE BER study. During the experimental process, in which the 58 trial cases were transcribed, no changes were made to the ETL vocabulary.

## 3.2 Abstraction in SIROCCO

Although Fact Primitives are specified at the most detailed level in the Fact Chronologies, SIROCCO is able to reason with the primitives at a more abstract level using an *Action/Event Hierarchy*. The Action/Event Hierarchy is a characterization and abstraction of the most-important actions and events that occur in engineering ethics scenarios. Cases may potentially be retrieved and matched based on similarity at higher levels of the hierarchy. This abstract matching capability is one of the aspects of SIROCCO that distinguishes it from similar CBR structural mapping programs, such as GREBE [Branting, 1991, 2000][4].

A portion of SIROCCO's Action/Event Hierarchy is shown in Figure 5.

[4] On the other hand, at least one CBR program that performs structural mapping, CaPER [Kettler *et al.*, 1994], also provides a form of abstract matching. However, CaPER does not support inexact matches between elements, does not focus on critical elements, and does not have well-defined temporal elements, as does SIROCCO.

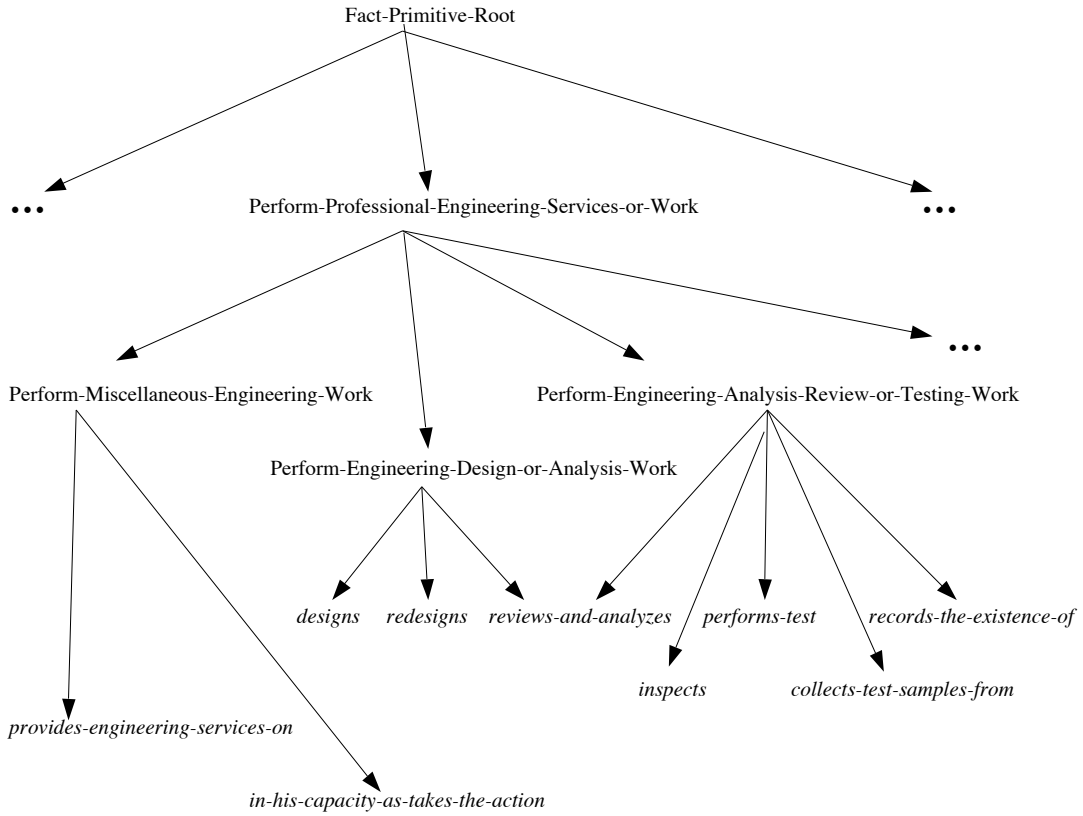



**Figure 5**: A Portion of SIROCCO's Action/Event Hierarchy

Fact Primitives are displayed in italics as the leaves of the hierarchy. Abstract categories are the inner nodes of the hierarchy. The depth of the hierarchy ranges from 2 to 5 levels and there are a total of 33 first level abstraction categories (i.e., those categories directly beneath the Fact-Primitive-Root). Note that the Action/Event Hierarchy is not a strict tree; for instance, the Fact primitive "reviews-and-analyzes" has two parents.

The 75 NSPE codes are also cast in an abstraction hierarchy, the *Code Hierarchy*, which groups related codes together according to similarity of the issues they address and was adapted from an abstraction of the codes found in an NSPE document [NSPE, 1996]. The Code Hierarchy is used by SIROCCO to assess the citation overlap of possible case citations to target cases, and it also provides important similarity information used to quantify SIROCCO's accuracy and to compare SIROCCO to the other methods in the experiments. In essence, the Code Hierarchy provides a means for assessing how well codes suggested by SIROCCO match up to the experts' suggestions and to the suggestions made by other programs. For instance, two different codes found in the same abstract code category are considered an inexact match for purposes of the experiments reported in this paper. A portion of the Code Hierarchy is shown in Figure 6.

Code provisions are found at the leaves of the hierarchy. Abstract categories are the inner nodes of the hierarchy. The hierarchy has 31 abstraction categories, covering all 75 NSPE BER codes, and the maximum depth of the hierarchy is 4.

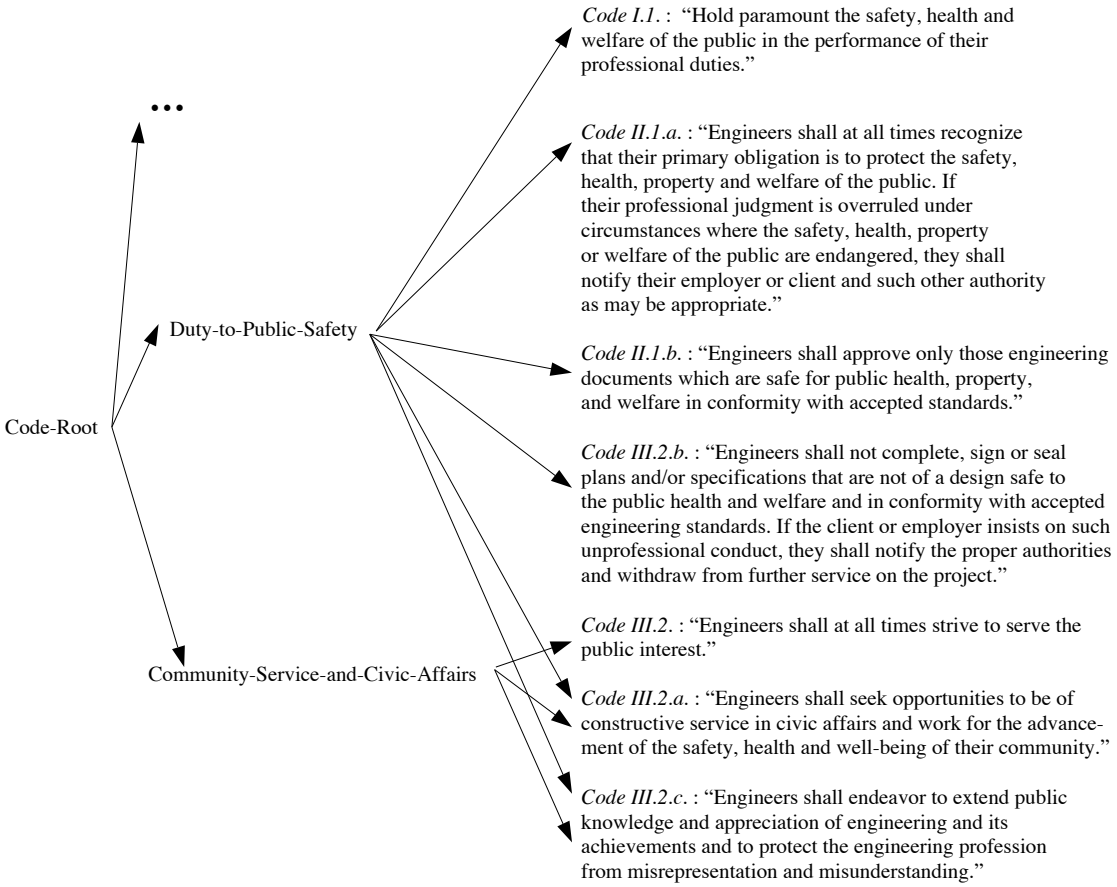



**Figure 6**: A Portion of SIROCCO's Code Hierarchy

## 3.3 Representing Operationalizations in Source Cases: The Extended Ethics Transcription Language

We have seen how to represent the facts of a case using ETL. Now let us turn our attention to the way operationalizations of ethics principles are represented in source cases. Entering a source case in the case base requires full specification of the case's Fact Chronology, as discussed in Section 3.1, plus a representation of the board's analysis of the case using the Extended Ethics Transcription Language (EETL). EETL represents the protagonist whose action is questioned, a conclusion about that action (e.g., ethical, unethical, or unknown), and justifications for that conclusion. EETL models arguments as a set of operationalizations, some of which support the

conclusion reached in the scenario, some of which conflict with the conclusion, and some of which provide relevant background information.

***The Board's Analysis.***

| | |
|---|---|
| ***Questioned Fact(s):*** | Fact 10 |
| ***Questioned Actor or Actors:*** | Engineer A |
| ***The Board's Conclusion:*** | Unethical |

***Table 1: The board cited the following evidence in support of their conclusion:***

| Code | Code Status | How Cited | Grouped With | Over rides | Why Relevant? | Why Violated, Not Violated, Changed, or Not Applicable? |
|---|---|---|---|---|---|---|
| I.1 | Violated | Explicitly discussed | None | II.1.c. | Engineer is involved in a professional situation in which a safety, health or welfare issue is at stake [1, 2, 4, 5] | Engineer's action does not hold paramount the safety, health, and welfare of the public [10] |
| ... | ... | ... | ... | ... | ... | ... |
| **Case** | **Citation Type** | **How Cited** | **Grouped with** | **Q #** | **Why Relevant?** | **Why Distinguished or Analogous?** |
| 84-5 | Analogous Precedent | Explicitly discussed | None | 1 | A client overrules Engineer's judgment [1, 3] Engineer complies with the client's judgment [10] | Engineer's complicity could result in a potentially dangerous situation [2, 5] |

***Table 2: The board cited the following evidence that <u>conflicts</u> with their conclusion:***

| Code | Code Status | How Cited | Grouped With | Over rides | Why Relevant? | Why Violated, Not Violated, Changed, or Not Applicable? |
|---|---|---|---|---|---|---|
| II.1.c | Not violated | Explicitly discussed | None | None | Engineer has a client [1] Engineer obtains confidential facts, data, or information through work for the client. [4, 5] | Engineer does not reveal confidential facts, data, or information to unauthorized parties [10] |
| ... | ... | ... | ... | ... | ... | ... |

***Table 3: The board cited the following information that neither directly supports nor conflicts with the conclusion:***

| Case | Citation Type | How Cited | Grouped with | Q # | Why Relevant? | Why Distinguished or Analogous? |
|---|---|---|---|---|---|---|
| 87-2 | Relevant, But Not Controlling | Explicitly discussed | 85-4, 82-2 | 1 | Engineer has a client [1] Engineer obtains confidential facts, data, or information through work for the client. [4, 5] | NA |
| ... | ... | ... | ... | ... | ... | ... |

**Figure 7**: EETL Tables Representing BER Analysis of Case 89-7-1 (excerpts)

Excerpts from the EETL representation of the board's analysis of Case 89-7-1 are depicted in Figure 7. As shown at the top of the figure, the board concluded that Engineer A acted unethically by not alerting the appropriate authorities of the safety problems he discovered during his engineering assignment. Engineer A's questioned action is represented by Fact 10 in the Fact Chronology, and a link to that Fact is provided.

The three tables of Figure 7 include codes and past cases that support the board's conclusion (Table 1), that conflict with the board's conclusion (Table 2), and that the board deemed otherwise relevant to the present case (Table 3). At the top of Table 1, the heading "Code" indicates a code cited by the board, here NSPE Code I.1, the public safety code discussed earlier. "Code Status" indicates whether the cited code was violated or not in this instance. Here, the board said that Code I.1 was violated. "How Cited" indicates whether the code or case was cited explicitly or implicitly. "Grouped With" indicates other codes and cases that were discussed together with the cited code or case. "Overrides" shows whether the cited code overrides another cited code in this instance. Here, Code I.1 was found to override Code II.1.c. "Why Relevant?" provides a rationale for the citation and links the cited code to specific steps of the Fact Chronology. The last "Why…" column provides a rationale for the board's conclusion with respect to this cited code and also links the code to the relevant steps of the Fact Chronology.

In other words, the top of Table 1 records the information that NSPE Code I.1 was, according to the board, violated in this instance and thus supports the conclusion that Engineer A acted unethically. Conversely, Table 2 records that Engineer A's actions were relevant to but did not violate the confidentiality code (NSPE Code II.1.c), thus providing evidence that the engineer acted ethically. Table 1 shows that Code 1.1 overrides II.1.c. The tables also record the specific steps in the Fact Chronology that explain the board's conclusions.

Figure 7 also shows that the board cited two cases (at the bottom of Table 1 and in Table 3.) One case, 84-5, is cited by the board as an analogous precedent, supporting the board's conclusion that the engineer acted unethically. Another case, 87-2, has some similarity to the present case and is thus deemed relevant (but not "controlling"). Here, "Case" indicates the case number cited by the board. "Citation Type" indicates whether the cited case was determined to be analogous or simply relevant. "Q#" refers to the relevant question number of the cited case's fact situation. Again, the "Why…" columns provide links from the specific steps in the source case's Fact Chronology that explain why the cited case was relevant or that distinguished the cited from the source case.

Information in the three tables of Figure 7 provides the basis for the operationalization techniques discussed earlier (Figure 1). For instance, *Principle Instantiations* and *Case Instantiations* link codes and past cases with the critical facts of the case and the temporal sequence

of those facts. These techniques are the most critical in providing extensional definitions of codes and past cases and in allowing SIROCCO to later reuse the definitions. The numbers in brackets under the "Why Relevant?" and "Why Violated …?" / "Why Distinguished …" headings of Figure 7 link a code or case to the facts of Case 89-7-1 (Figure 3). For instance, Facts 1, 2, 4, and 5 (and their temporal sequence) of Case 89-7-1 provide an extensional explanation of Code I.1's relevance to Case 89-7-1, while Fact 10 provides an extensional explanation of Code I.1's violation. These links help SIROCCO access the extensionally-defined codes and past cases when analyzing new cases.

While instantiations are critical in the extensional definition of codes and cases, other techniques also make a contribution. *Principle Grouping* and *Case Grouping* extensionally group codes and cases and allow SIROCCO to explore related codes and cases. This information is denoted under the "Grouped With" heading of Figure 7. For instance, in their analysis of Case 89-7-1, the board grouped Cases 85-4 and 82-2 with Case 87-2. *Operationalization Reuse* is a meta-technique of sorts; it is used by the board to reuse any of the other 8 techniques in the context of a new case. This technique would be employed, for instance, when the board chooses to reuse one of the principle instantiations in Figure 7 in the analysis of a new case.

The above operationalization techniques (i.e., *Principle Instantiation*, *Principle Grouping*, *Case Instantiation*, *Case Grouping*, and *Operationalization Reuse* from Figure 1) contribute most directly to the retrieval of relevant cases and codes and are the ones tested in the experiment described in this paper. The remaining techniques provide various types of explanatory information. For instance, *Conflicting Principles Resolution* is useful for indicating when two or more of the retrieved codes are in conflict with one another where one code takes precedence over the other(s). In Case 89-7-1, Code I.1 overrides Code II.1.c; the "Overrides" column of Table 1, Figure 7 indicates this. This information could subsequently be used to explain a conflict between the same codes in the context of another case.

A more detailed description of the operationalization techniques, including excerpts of their application in actual NSPE BER cases, can be found in [McLaren, 1999, Chapter 2].

## 3.4 SIROCCO's Case Base

SIROCCO's case base consists of a subset of the NSPE BER cases that were analyzed during the domain study. As mentioned above, 184 foundational cases, covering 135 fact situations[5] and culled from the 475 cases decided by the NSPE BER between 1958 and 1992, are included in

[5] A case is defined as a Fact Chronology together with a chosen Questioned Fact from that chronology. Thus, if a scenario raises multiple ethical questions, one Fact Chronology can represent multiple cases.

SIROCCO's case base and were used to design, implement, and refine the program. The foundational cases cover a reasonable number of important ethics topics but also provide limited coverage of cases outside of those topics. More specifically, 135 cases cite at least one code related to the *Selected Topics* (see Table 1). The remaining 49 cases do not cite any codes from the Selected Topics but cite at least one code from the *Non-Selected Topics*. The cases are widely spread across these topics and cite virtually all of the 75 NSPE codes.

| Category | Types of Ethical Topics in the Category |
|---|---|
| **Selected Topics** | Public safety, confidential information, duty to employer, credit for engineering work, proprietary interests, and honesty in reports and public statements. |
| **Non-Selected Topics** | Conflicts of interest, honesty in advertising, criticizing other engineers, competence and qualifications, … |

**Table 1**: Topics Addressed within the NSPE BER Cases and Represented in SIROCCO's Case Base

Twelve independent case enterers transcribed the foundational cases into EETL using the case-acquisition web site. Most of the foundational cases provided by the case enterers were at least slightly amended to correct issues such as poorly constructed Facts and incorrect usage of Fact Primitives. However, in most instances, the critical aspects of the representations provided by the case enterers were not altered. The case enterers reported that it took, on average, 2 to 3 hours to transcribe a single source case (i.e., representing both a Fact Chronology and the board's analysis). Most of this time, they reported, was used to transcribe each case's analysis. Translating textual case facts to the Fact Chronology (expressed in ETL, as in Figure 3) took much less time. Thus, it is considerably easier to provide target cases for SIROCCO to analyze than it is to provide new source cases for its case base.

## 3.5 SIROCCO's Retrieval and Analysis Process

As shown in Figure 8, SIROCCO retrieves cases in two stages and uses structural mapping in the second stage. In this respect SIROCCO is similar to analogical retrieval programs [see, e.g., Thagard *et al.*, 1990; Branting, 1991, 2000; Forbus *et al.*, 1994].

On the left of the figure are the algorithm's steps. The right side shows particular knowledge sources, namely the case base and information about Principle and Case operationalizations. Case operationalizations are represented, at least partially, as references from one case to another. Principle operationalizations are references from cases to codes. The case base and the operationalizations are used in all three stages of the algorithm, but the relevant information is passed from the first stage to the latter stages.

In the first stage the program rapidly retrieves a set of source cases that match the target case, at least superficially. It achieves this by matching the Fact Primitives of the target case, and abstractions of those primitives, to all of the source cases and returning the best *N* matches. Stage 1 is superficial in the sense that it doesn't account for case structure – that is, the relationship between and within facts – but it does account for abstract Fact Primitive matches and matches to important source case Facts.

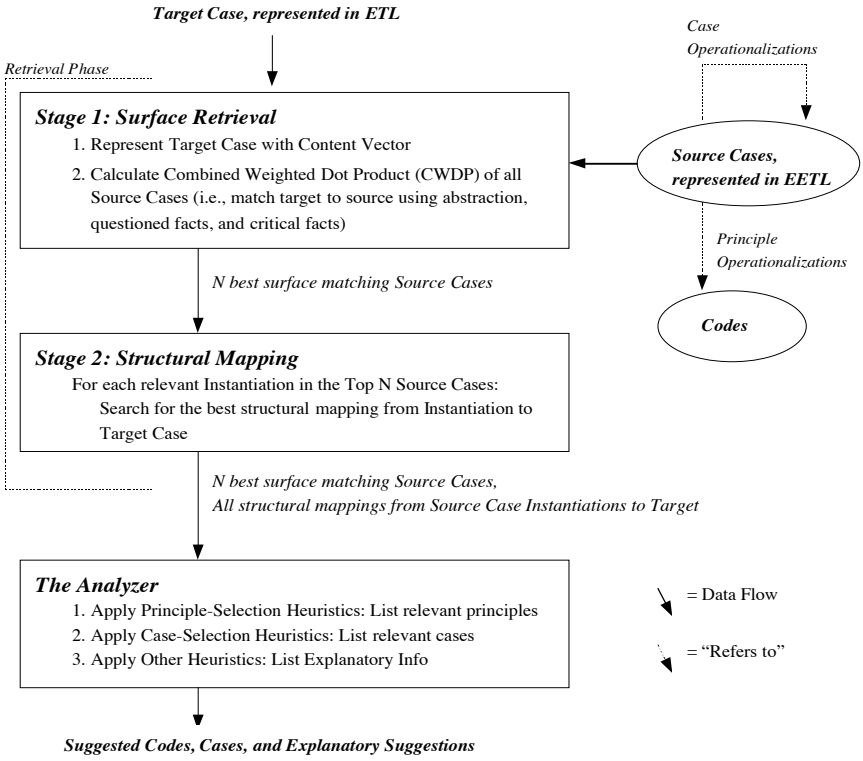



**Figure 8**: SIROCCO's Algorithm.

In the second stage a deeper, but more expensive *structural mapping* is executed between the target and the best *N* matches from stage 1. The structural mapping does account for case structure by checking for similar types of Fact Primitives, consistency of actors and objects, and consistency of temporal relations between the target and source cases. The program uses A* search in stage 2 to find the best structural matches [Branting, 1991, 2000] and applies Gentner's structural consistency constraints (i.e., one-to-one mapping and parallel connectivity) [1983].

*Principle Instantiations* and *Case Instantiations* are critical to the efficiency and accuracy of both stages of the retrieval algorithm. Stage 1's accuracy improves because more weight is given to matches to the instantiations' Fact Primitives. Stage 2 is made both more efficient and more accurate because its structural mapping routine is focused on a subset of each source case's Fact

Chronology (i.e., the subset associated with an instantiation). This reduces the search space and enables the algorithm to focus on those facts that are most critical to the ethical evaluation of a case.

### 3.5.1 Stage 1 of SIROCCO: Surface Retrieval

Stage 1 uses *content vectors* [Forbus *et al.*, 1994] to compare each new target problem to all of the source cases. Comparison is performed at each of four predefined and increasingly more abstract levels of the Action/Event Hierarchy, shown in Figure 9, and the results are combined. By predefining the abstraction levels and pre-storing each source case's information at each of those levels, the similarity computation is relatively fast.

| | |
|---|---|
| ***Level 1 - Fact Primitive:*** | Exact matches between Fact Primitives. |
| ***Level 2 - Fact Group :*** | Matches between Fact Primitives that share the same parent abstraction category (i.e., matches one level up the Action/Event Hierarchy). |
| ***Level 3 - Sibling Group:*** | Matches between Fact Primitives that share the same abstraction category two levels up the Action/Event Hierarchy. |
| ***Level 4 - Root Group:*** | Matches between Fact Primitives at the "root group" level (i.e., one level below the root of the Action/Event Hierarchy). |

**Figure 9**: SIROCCO's Predefined Abstraction Levels for Matching Fact Primitives

Figure 10 shows the content vectors for levels 1 and 2 of Case 89-7-1. To simplify the figure, the content vectors generated for levels 3 and 4 are not shown. Each vector specifies the Fact Primitives (or abstractions) and a count of how many times each appears. The asterisks ("***") indicate the questioned fact in the chronology. Compare the Level 1 (Fact Primitive) Content Vector to the Fact Primitives that appear in Case 89-7-1 of Figure 3.

Stage 1 uses these content vectors to compute a *combined weighted dot product* (CWDP) for every source case in the case base. It then provides the *N* best source cases with respect to the CWDP to Stage 2 of the algorithm. The CWDP measures how well a target case matches a source case at various levels of abstraction and in terms of certain critical facts: the questioned fact and facts deemed critical by virtue of their connection to source instantiations. Weights that are provided to SIROCCO as parameters are applied to matches at the four abstraction levels and to matches of a source case's critical and questioned facts.

| Level 1 (Fact Primitive) Content Vector: | Level 2 (Fact Group) Content Vector: |
|---|---|
| (Hires-the-Services-of 1) | (Work-as-an-Employed-or-Contract-Professional-Engineer 1) |
| (May-be-Hazardous-to-Safety 1) | (Deal-with-Potential-Dangers-or-Hazards 1) |
| (Instructs-to 1) | (Order-Subordinate-to-Perform-Task 1) |
| (Inspects 1) | (Perform-Engineering-Analysis-Review-or-Testing-Work 1) |
| (Discovers-That 1)<br>(Knows 1) | (Know-or-Believe-Something 2) |
| (Informs-That 2) *** | (Disclose-Information 2) *** |
| (Writes-Paper/Article 1) | (Write-an-Engineering-Related-Document-Article-or-Paper 1) |
| (Provides-Limited-Information-to-Regarding 1) | (Withhold-Information 1) |

**Figure 10**: Content Vectors for Case 89-7-1

The specific formula for the CWDP is as follows. Given a target case (T) and a source case ($S_n$), the CWDP for $S_n$ is:

$$CWDP = WDP + (QFW_x * MWDP) + (CFW_y * MWDP) \quad (\mathbf{1})$$

$$WDP = (W_1 * DP_1 / MDP_1) + (W_2 * DP_2 / MDP_2) + (W_3 * DP_3 / MDP_3) + (W_4 * DP_4 / MDP_4) \quad (\mathbf{2})$$

Where:

WDP is the weighted dot product; 0 <= WDP <= 1.0

QFWx ≤ 1.0 and is the questioned fact weight at the most specific match level.

CFWy ≤ 1.0 and is the critical fact weight at the most specific match level.

MWDP is the maximum weighted dot product (WDP) over all source cases.

$W_1 + W_2 + W_3 + W_4 = 1.0$ (Pre-defined weights corresponding to each abstraction level).

$DP_1 \ldots DP_4$ are the normalized dot products of T and Sn at each abstraction level (i.e., levels 1 through 4).

$MDP_1 \ldots MDP_4$ are the maximum dot products at each abstraction level over all source cases.

The dot products (i.e., $DP_1 \ldots DP_4$) are calculated as in vector arithmetic, by summing the product of matching vector elements (i.e., matching facts). That is, if both the target and source share a particular vector element at a particular abstraction level (e.g., "Hires-the-Services-of" at level 1), then the product of the count of how many times each appears is calculated (e.g., If "Hires-the-Services-of" occurs 2 times in the source and 1 time in the target, the product for this element would be 2; if "Hires-the-Services-of" occurs 2 times in the source but not in the target, the product for this element would be 0.). The sum of all the matching element products is the dot product at that level of abstraction. Since this calculation tends to favor long fact chronologies, the dot products are normalized by dividing by the sum of the content vector elements, which is roughly equal to the length of a fact chronology. As shown in equation 2, each dot product is also

normalized by dividing by the maximum dot product ($MDP_1$ … $MDP_4$) over all source cases at the corresponding abstraction level.

Equation 2 captures how well the facts of two cases match both exactly and at more abstract levels in the Action/Event Hierarchy. The contribution of an exact match is more than that of an abstract match, since the default weights applied by SIROCCO[6] are $W_1 = 0.53$, $W_2 = 0.27$, $W_3 = 0.14$, and $W_4 = 0.06$.

The latter two addends of equation 1 account for matching of questioned and critical facts, respectively, between the target and source case. Each of these addends is equivalent to the best WDP over all source cases (i.e., MWDP), times a weight that corresponds to the abstraction level of the match ($QFW_x$ for questioned facts, $CFW_y$ for critical facts). Questioned facts are considered more important than critical facts. This is reflected in the default abstraction-level weights: 1.0, 0.5, 0.25, and 0.125 for questioned facts, 0.333, 0.111, 0.036, and 0.012 for critical facts. Matching a questioned fact can have a dramatic effect; the weighting factor is applied to the *highest* WDP over *all* source cases.

A brief and partial example serves to illustrate the calculation of Equations 1 and 2. Given the Level 1 (Fact Primitive) content vector of Case 89-7-1, shown in Figure 10, together with the Level 1 content vector of Case 76-4-1, the dot product of these two cases at this level is shown in Figure 11. Four facts match between the two cases, with a count of 1 between two of the matching facts and counts of greater than 1 for the other two matching facts. The dot product is 9, as shown in the lower right-hand corner of the figure. The normalized version of the dot product ($DP_1$ in equation 2) is 9 / 12, since 76-4-1 (the source case) has a *total* count of 12. $DP_1$ is then normalized again by dividing by the largest dot product found at this abstraction level ($MDP_1$), and is multiplied by 0.53, the default weight at level 1 ($W_1$). A similar calculation occurs at the other three abstraction levels, leading to the WDP computation of equation 2. The combined weighted dot product (CWDP, equation 1) is then calculated by summing the WDP together with the questioned fact addend (a value greater than 0, as the questioned facts of the two cases match, see "Informs-That" in Figure 11) and the critical fact addend (also a value greater than 0, as at least one fact of 89-7-1 matches a critical fact of 76-4-1, e.g., "Discovers-That").

[6] Although weights may be specified by the user, in practice and in the experiments reported in this paper, default weights provided the best results in informal experimentation. In general, the specific weights did not appear to make much difference in the informal experiments I conducted, as long as a similar order of magnitude relationship existed between the different levels.

| Case 89-7-1 Level 1 Content Vector: | Case 76-4-1 Level 1 Content Vector: | Dot Product |
|---|---|---|
| (Hires-the-Services-of 1) | (Hires-the-Services-Of 1) | 1 |
| (May-be-Hazardous-to-Safety 1) | --- | 0 |
| (Instructs-to 1) | (Instructs-to 1) | 1 |
| (Inspects 1) | --- | 0 |
| (Discovers-That 1) | (Discovers-That 3) | 3 |
| (Knows 1) | --- | 0 |
| (Informs-That 2) *** | (Informs-That 2) *** | 4 |
| (Writes-Paper/Article 1) | --- | 0 |
| (Provides-Limited-Information-to-Regarding 1) | --- | 0 |
| --- | (Reviews-and-Analyzes 1) | 0 |
| --- | (Terminates-the-Services-of 1) | 0 |
| --- | (Pays-For 1) | 0 |
| --- | (Calls-a-Hearing-Regarding 1) | 0 |
| --- | (Claims-That 1) | 0 |
| Total Element Count: 10 | Total Element Count: 12 | **9** |

**Figure 11**: Dot Product calculation for Level 1 of Cases 89-7-1 and 76-4-1

### 3.5.2 Stage 2 of SIROCCO: Structural Mapping

Stage 2 uses A* search to find structural mappings between the target case and the best *N* source cases from Stage 1. SIROCCO's use of A* search improves upon that of GREBE [Branting, 1991, 2000] by taking temporal relations into account, by supporting abstract matches, and by accommodating a wider range of factual scenarios. The goal is to map each of the Facts of a source instantiation to a corresponding Fact in the target case while maintaining a one-to-one and consistent mapping between their Actors and Objects.

The initial node of the search space maps the source's questioned Actor to the target's questioned Actor. Subsequent nodes are generated by selecting an unmapped Fact from the source and mapping it to each of the unmapped Facts in the target for which the source and target Fact Primitives match either exactly or abstractly. Each subsequent node corresponds to:

1. a tentative mapping between the source and target Facts at the same level of abstraction,
2. a one-to-one Fact mapping between the source and target (i.e., no Fact is mapped more than once),
3. a one-to-one, consistent set of Actor and Object mappings entailed by the Fact mappings, and
4. consistent temporal relations between mapped Facts of the source and target. (Temporal relations are consistent if the Allen relations of every pair of mapped source Facts intersect with the Allen relations of the corresponding pair of target facts. Note that

GREBE did not account for temporal constraints at all, meaning that facts in completely different time order between the source and target could match.)

Note that the above conditions satisfy Gentner's structural consistency constraints [1983].

An "empty" node is also generated at each ply of the search to represent a Fact mapping failure but allowing for subsequent Fact mappings along this path. The goal node is reached when the current depth equals a pre-defined solution depth (equal to the number of Facts to match in the source) and either the current node has the lowest mismatch score of all open nodes, as defined by the A* cost function, or the list of nodes is empty.

The standard A* cost function is applied to evaluate each node. The cost function combines two measures: (1) the quality of the partial solution up to the current node and (2) an estimate of the cost of achieving a solution from the current node.

$$f(n) = g(n) + h'(n)$$

Where $n$ is a node at depth d in the search tree
$g(n)$ is a measure of the degree of mismatched Facts at d
$h'(n)$ is the best possible score to complete the mapping

More specifically, $g(n)$ is equal to the *mismatch cost* at $n$ divided by d. The mismatch cost is a summation of the degree of mismatch at each node up to and including $n$. The mismatch costs at different levels of abstraction are: 0 for an exact match (i.e., level 1), 0.4 for a match at level 2 (Fact Group), 0.6 at the level 3 (Sibling Group), 0.9 at level 4 (Fact Root), and 1.0 for a "failed" match[7]. Thus, for a search path at depth = 2 in which a Fact Primitive match and a Fact Group match have been proposed, the mismatch cost is 0.4 (i.e., 0 + 0.4) and $g(n)$ = 0.2 (0.4 / 2). Matching at these different levels of abstraction allows SIROCCO, unlike GREBE, to be sensitive to and extend paths during A* search that exhibit less than precise matches between source and target cases.

The $h'(n)$ function is calculated by dividing the mismatch cost at n by the fixed solution depth. The solution depth is always fixed to be the number of Facts in the instantiation (even if a Fact doesn't match, an "empty" match node is created). Thus, $h'(n)$ provides the mismatch cost that would be attained by achieving an exact match (i.e., mismatch = 0) at each node from $n$ until the goal node is reached. For instance, using the example from the previous paragraph and assuming a fixed depth of 4 (i.e., the source instantiation has 4 Facts), $h'(n)$ = 0.1 (0.4 / 4). Because $h'(n)$ is the most-optimistic possible completion of the mapping, it satisfies the *admissibility* condition [Ginsberg, 1993, p. 78] and guarantees that the algorithm will never return a sub-optimal goal node. In particular, SIROCCO always returns the minimum $f(n)$ found at the fixed solution depth.

SIROCCO's search is actually a variant of A* because the solution depth is fixed; it is always equal to the number of Facts in the instantiation, rather than a variable, minimum depth, as in standard A*. The solution depth is fixed because the "empty" node generated at each level acts as a catch-all; even if no successful mapping occurs at that level the current path may be continued with this level counted as a "failed" match (i.e., a 1.0 mismatch cost).

Here's a simple example of SIROCCO's A* structural mapping. One source case in SIROCCO's case base that is similar to 89-7-1 – and is passed to Stage 2 by Stage 1 of SIROCCO's retrieval algorithm – is Case 76-4-1. In 76-4-1 a firm hired an engineer to study the effect of its planned discharges on water quality, but does not like the results of the study, terminates the engineer's consultation and directs him not to disclose the results to anyone or to write a report on those results. When the engineer discovers that the firm has presented contrary evidence at a regulatory hearing, he does not disclose the results of his study. The board cites Code III.2.b, the principle mentioned earlier dealing with an engineer's obligation not to prepare unsafe specifications, in their analysis, thus linking it as an instantiation to the facts in Case 76-4-1.

Figure 12 depicts the goal node generated by SIROCCO's A* search while mapping that instantiation to example target Case 89-7-1. This mapping satisfies the four conditions mentioned above; namely, a one-to-one mapping between Fact Primitives at the same level of abstraction that also preserves consistent Actor, Object, and temporal mappings. SIROCCO has succeeded in mapping Case 89-7-1 to four of the five facts of 76-4-1's instantiation of Code III.2.b. Three Fact Primitives of the instantiation match precisely to the facts of Case 89-7-1, one matches at the Fact Group level, and one does not match at all. Since this is the goal node, the current depth and solution depth are both equal to 5 (the total number of facts in the instantiation) and the cost function ($f(n)$) has a value of (1.4 / 5) + (1.4 / 5) = 0.56. It is usually more convenient to think of $f(n)$ in terms of a *match percentage*. Because 0.56 measures mismatch and the maximum possible mismatch is 2.0, the match percentage of an instantiation is (2.0 – $f(n)$) / 2.0. Thus, the match percentage of the node in Figure 12 is 1.44 / 2.0 = 72%.

[7] Since Fact Primitives can have multiple parents (e.g., "reviews-and-analyzes" in Figure 5), there is sometimes a choice of which abstraction to use for the mistmatch cost. SIROCCO prefers the minimum graph distance (e.g., preference for a sibling rather than a cousin).

| Mapping Level | Facts of the Target Case (Case 89-7-1) | Facts of Source Case Inst. III.2.b. (Case 76-4-1) |
|---|---|---|
| *Exact Match* | 1. Client X **<hires-the-services-of>** Engineer A | 1. XYZ Corporation **<hires the services of>** Engineer Doe **<for>** (Engineer Doe **<reviews and analyzes>** Discharge). |
| *Fact Group level* | 4. Engineer A **<inspects>** Apartment House P. | 2. Engineer Doe **<reviews and analyzes>** Discharge |
| *Exact Match* | 5. Engineer A **<discovers that>** (Apartment House P **<fails standards and may be hazardous to safety>**). | 3. Engineer Doe **<discovers that>** (Discharge **<fails standards and may be hazardous to safety>**). |
| *No Match* | ***None*** | 6. XYZ Corporation **<instructs>** Engineer Doe **<to>** (Engineer Doe **<does not write paper/article>** …) |
| *Exact Match* | 10. Engineer A **<does not inform>** Any Third Parties **<that>** (Apartment House P **<fails standards and may be hazardous to safety>**). | 11. Engineer Doe **<does not inform>** Control Authority **<that>** (Discharge **<fails standards and may be hazardous to safety>**). |

**Figure 12**: Mapping of Principle Instantiation III.2.b of Case 76-4-1 to Case 89-7-1

While clearly less than a perfect match, this level of match percentage is high enough for SIROCCO to suggest this code as potentially relevant to Case 89-7-1. Interestingly, while public safety code III.2.b[8] clearly has relevance to the example case, it was not cited by the board. Thus, this is an example of how SIROCCO is able to find relevant codes that even the experts did not find.

### 3.5.3 The Analysis Stage

In the Analysis stage, SIROCCO combines the results of multiple source cases to generate suggestions, rather than relying on a single best match. As output, SIROCCO lists possibly relevant codes and cases and can also generate, on request, heuristic reasons for its suggestions, additional relevant suggestions, and selected structural mappings between the target and select instantiations, similar to what is shown in Figure 12.

An abridged version of the program's output for Case 89-7-1 is shown in Figure 13. It shows all of the source cases and a portion of the codes and other suggestions that SIROCCO generates for this case. It also shows SIROCCO's reasons for selecting Code III.2.b. and Case 76-4-1, based on the program's selection heuristics. Finally, the arrows and corresponding text in the figure indicate where four of the operationalization techniques were used to help SIROCCO make its suggestions.

The code and case suggestions are made by selection heuristics that leverage the core operationalization techniques (i.e., *Principle Instantiation*, *Principle Grouping*, *Case Instantiation*, *Case Grouping*, and *Operationalization Reuse*). The heuristics favor codes that, for example, (1)

occur more frequently in the top-ranked cases of Stage 1, (2) match a high percentage of critical facts in cases citing the code, or (3) are grouped with other codes cited in those cases.

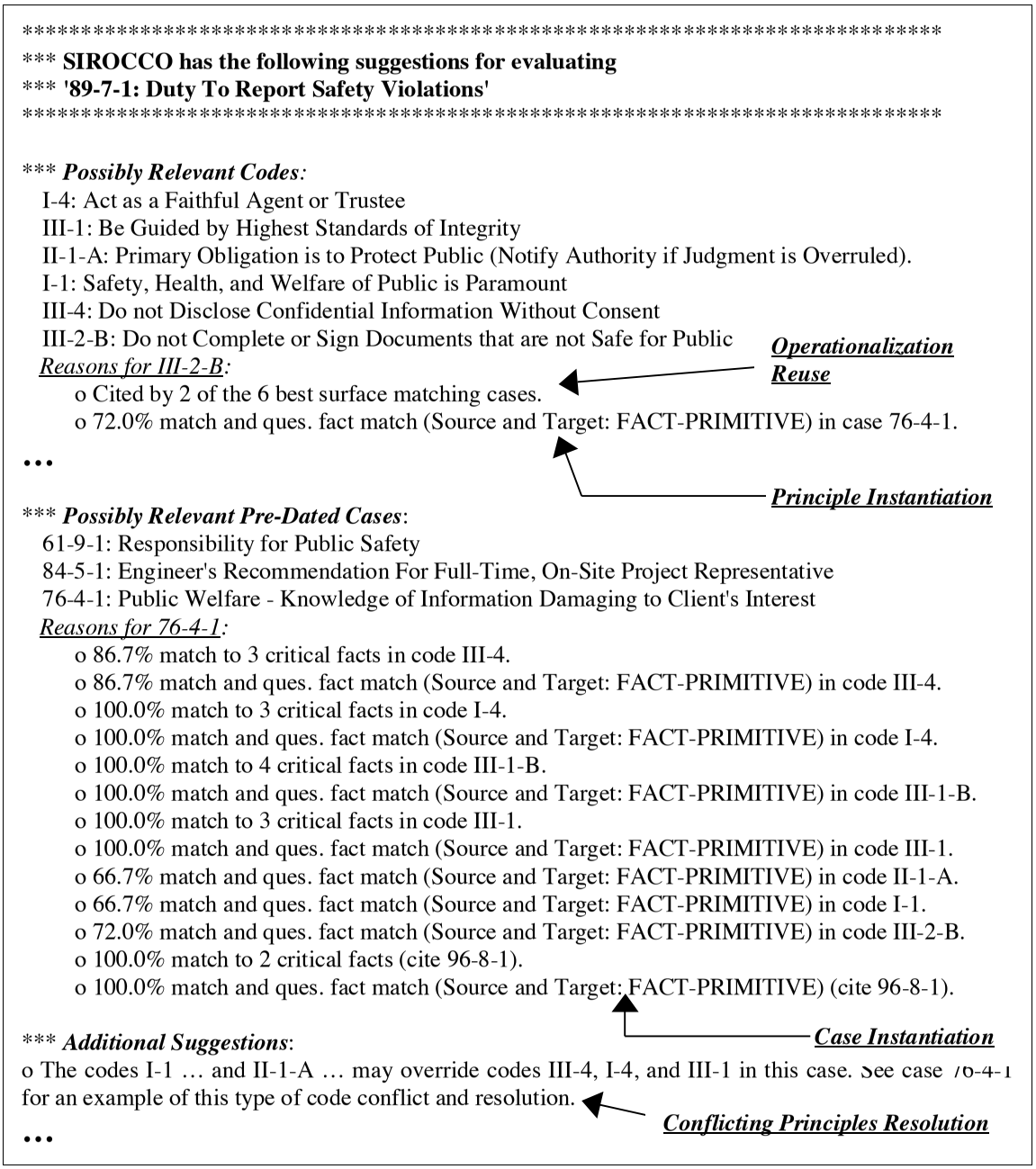



**Figure 13**: Excerpts of SIROCCO's output for Case 89-7-1

For example, the program suggests Code III.2.b. as potentially relevant for two reasons, both of which are shown in Figure 13. The first reason is that two of the top-rated cases from Stage 1 cited this code. As indicated in the figure, this is an example of *Operationalization Reuse*, as the

[8] "Engineers shall not complete, sign or seal plans and/or specifications that are not of a design safe to the public health and welfare and in conformity with accepted engineering standards."

program leverages and reuses past citations (i.e., operationalizations) to select Code III.2.b. in the current case analysis. The second reason that Code III.2.b. is suggested as potentially relevant is because the structural mapping of Case 89-7-1 to the instantiation of Code III.2.b of Case 76-4-1 (see Figure 12) is sufficiently strong. As indicated in Figure 13, this is an example of the use of a *Principle Instantiation*, as the program draws an analogy to a specific past instantiation of Code III.2.b. Similar heuristics are used to select cases, as can be seen by the reasons for the selection of Case 76-4-1 in Figure 13. For instance, a *Case Operationalization* supplies one of the reasons for this selection, as the program draws an analogy to a specific past instantiation of Case 76-4-1. This is indicated at the bottom of Figure 13.

The non-core operationalization techniques (i.e., *Fact Hypotheses*, *Principle Revision*, *Conflicting Principles Resolution*, and *Principle Elaboration*) allow SIROCCO to make additional suggestions. For each of the codes and cases it suggests as possibly relevant, SIROCCO attempts to find operationalized information from the previous cases that may be relevant and helpful in the present circumstances. For instance, as shown at the bottom of Figure 13, SIROCCO suggests that the codes dealing with public safety (I.1 and II.1.a) may override codes dealing with confidentiality (III.4), standards of integrity (III.1), and duty to an employer (I.4), because they did so in the context of Case 76-4-1. It makes this suggestion because 76-4-1 is an example of this conflict and all of the same codes are suggested by SIROCCO in this case. Notice that the board actually *did* employ the *Conflicting Principles Resolution* technique in Case 89-7-1. This can be seen in Table 1 of Figure 7 in the row beginning "I.1," which indicates that the public safety code overrides a code dealing with confidentiality (II.1.c). While the specific confidentiality code that is overridden is different, the conflict is essentially the same as that in Case 89-7-1.

# 4 The Experiments

In order to assess the contribution of the board's operationalized code provisions and cases, I conducted a series of experiments, including an ablation experiment. The experimental design informally followed that of Rissland *et al* [1997]. The 184 foundational cases served as SIROCCO's case base for the experiments; the 58 trial cases were provided as input target cases to test the program. Two independent case enterers were employed to transcribe all of the trial cases into extended ETL. Their transcriptions were submitted unaltered to SIROCCO for processing. The 58 trial cases were chosen from a set of 77 cases decided by the BER after 1993: 44 trial cases were chosen randomly from 52 Selected Topics cases and 14 trial cases were chosen randomly from 25 Non-Selected Topics cases (see Table 1).

SIROCCO's performance was compared with that of five other methods:

**RANDOM**: For each target case, selects codes/cases at random from all codes and cases in the case base.

**INFORMED-RANDOM**: For each target case, selects codes and cases at random from the most frequently cited codes and cases in the case base.

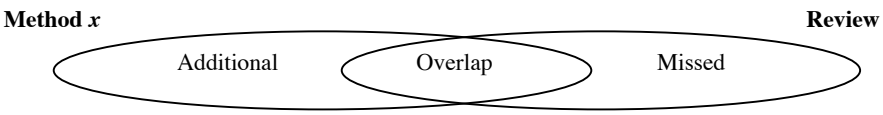
**NON-OP SIROCCO**: An ablated version of SIROCCO with almost no functionality related to operationalizations. Selects cases using SIROCCO's Stage 1 algorithm – with no Stage 2 and no operationalizations – and selects codes using only minimal operationalization information, in particular the *Operationalization Reuse* technique (i.e., selects codes that appear most frequently in the top *N* cases from Stage 1).

**MG (Managing Gigabytes):** A full-text retrieval method that converts a target ethics case into a term vector and then selects codes and cases by comparing the target term vector to the term vectors for all codes and all source cases [Witten *et al.*, 1999].

**EXTENDED-MG**: Like MG, but uses some operationalization information. In particular, selects cases using the term vector approach of MG but selects codes by "Operationalization Reuse" (i.e., selects codes that appear most frequently in the top *X* cases selected by term vector retrieval).

Most relevantly for the present paper, the comparison with NON-OP SIROCCO focuses on the contribution of operationalization information. In particular, it focuses on the core subset of the operationalization techniques in Figure 1, those which contribute directly to retrieving relevant cases and codes (i.e., *Principle Instantiation*, *Principle Grouping*, *Case Instantiation*, *Case Grouping*, and *Operationalization Reuse*). In NON-OP SIROCCO, this core-set of operationalization techniques was turned off. NON-OP SIROCCO did, however, prefer codes that appeared most frequently in the list of the *N* top-rated cases; this can be considered a weak kind of operationalization technique, but it is the only such information NON-OP SIROCCO used.

Each method, including SIROCCO, processed each of the trial cases one-by-one, and its retrieval results were compared to the BER's code and case citations for the same case. The *F-measure,* an information retrieval metric that combines precision and recall, was used to calculate the overlap between a method's solution and the board's solution [van Rijsbergen, 1979, p. 173-176; Lewis *et al.*, 1996]. A Venn diagram depicting the overlap and the equations for precision, recall, and the F-Measure is shown in Figure 14. (Note that ß was set to 1.0 to assign equal weights to precision and recall.)




**Method *x*** = The code and case suggestions produced by SIROCCO or one of the other methods.
**Review Board** = The code and case citations of the ethics review board.
**Overlap** = The code and case citations shared by both Method *x* and the review board.
**Additional** = The extra code and case suggestions made by Method *x* but not cited by the review board.
**Missed** = The set of code and case suggestions lacking in Method *x* but cited by the review board.

**Precision (*P*)** = Overlap / (Overlap + Additional)  **Recall (*R*)** = Overlap / (Overlap + Missed)

**F-Measure** (Method *x*, Review Board) = $\frac{\left(\beta^2 + 1\right)PR}{\beta^2 P + R}$  (Maximum F-Measure = 1.0)

**Figure 14**: Calculation of the F-Measure in the SIROCCO Experiments

For each trial case, two F-Measure values were calculated. One indicated the extent of exact matches of cited codes and cases between the two solutions. The other indicated the extent of inexact matches, combining inexact code matches with inexact case matches. Inexact code matches were determined using the Code Hierarchy. For instance, in Figure 6 Code I.1. and Code II.1.a would be considered an inexact code match, since they share the same parent node (i.e., Duty-to-Public-Safety). Inexact case matches were determined using a citation overlap metric that measures the inverse of the length of the citation path between two cases. For instance, if a case directly cites another, the overlap is 1/1. If two cases share a citation to a third case, the overlap is 1/2 [McLaren and Ashley, 1999].

# 5 Experimental Results

As can be seen in Figure 15, SIROCCO attained the highest mean F-Measure for both exact and inexact matching over the 58 trial cases. The closest competitor to SIROCCO for exact matching was EXTENDED-MG and the closest competitor for inexact matching was MG. NON-OP SIROCCO's mean F-Measure for exact matching was third best, while for inexact matching it was fourth best. Not surprisingly, both of the random selection methods (i.e., INFORMED-RANDOM and RANDOM), performed worse than any of the other methods, according to the mean F-Measure.

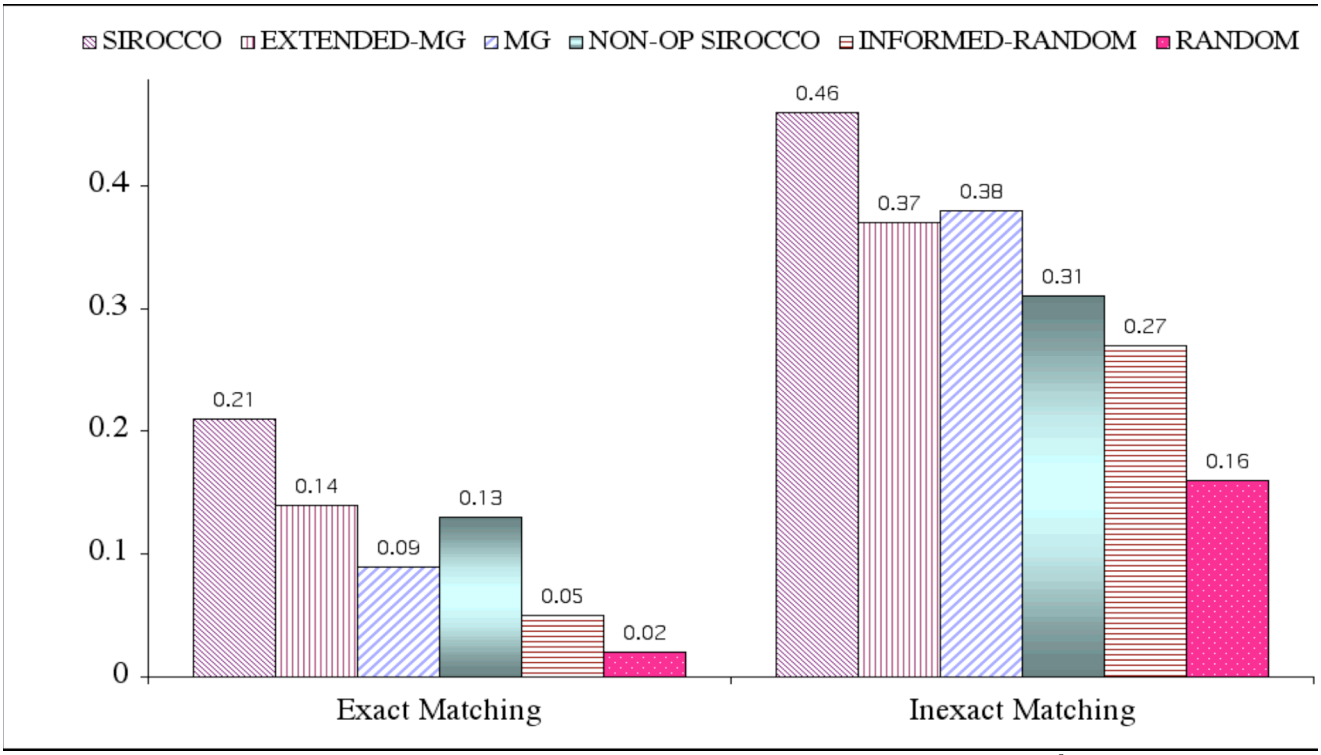



**Figure 15**: Mean F-Measures for all Methods over all of the Trial Cases[9]

Although the mean F-Measure is a useful approximation of the relative accuracy of SIROCCO versus the other methods, it does not say whether the differences between SIROCCO and the other methods are statistically significant. Since the data generated by benchmarking each method against the board's citations using the F-Measure was highly non-Gaussian, a *nonparametric bootstrap procedure*[10] was applied [Davison and Hinkley, 1997]. The results of the procedure applied to pairs of methods for exact matching and inexact matching are shown in Tables 2 and 3.

The information in the tables is interpreted as follows. Each row represents a pairwise comparison of two of the methods. The first column contains the compared methods. The goal was to compare SIROCCO to each of the other methods, so there are five rows in the table. The second and third columns show the results of the nonparametric bootstrap procedure applied to that comparison. The first value in the second column represents the 2,500$^{th}$ mean difference of Method 1 minus Method 2, while the second value in the second column represents the 97,500$^{th}$ mean difference. Because the 100,000 differences for each method are sorted, these two values demarcate the 95% Confidence Interval for mean difference. Roughly speaking, this is a range

[9] In [McLaren and Ashley, 2000] there is an error in this graph. The exact matching value for NON-OP SIROCCO (0.13) is as reported here and in [McLaren, 1999] rather than what is reported in [McLaren and Ashley, 2000] (0.1).

estimate of the average difference in F-Measure between Method 1 and Method 2 for exact matching. For instance, the average mean difference between SIROCCO and EXTENDED-MG, shown in row 1 of Table 2, is between +0.018 higher for SIROCCO and +0.121 higher for SIROCCO. The p-value is calculated by first subtracting the corresponding 100,000 means of the compared methods. Then the proportion of those calculations (x / 100,000) that are less than or equal to zero is multiplied by two. The resulting value is the probability of a mean difference = 0; in other words, the probability that Method 1 is no more accurate than Method 2, according to the F-Measure.

| **Methods Compared** | **95% Confidence Interval for mean difference** | **p-value for mean difference=0** |
|---|---|---|
| SIROCCO vs. EXTENDED-MG | (0.018, 0.121) | 0.008 |
| SIROCCO vs. MG | (0.074, 0.180) | < 0.001 |
| SIROCCO vs. NON-OP SIROCCO | (0.039, 0.126) | < 0.001 |
| SIROCCO vs. INFORMED-RANDOM | (0.116, 0.210) | < 0.001 |
| SIROCCO vs. RANDOM | (0.139, 0.240) | < 0.001 |

**Table 2**: Exact Matching Comparisons Using the Nonparametric Bootstrap

| **Methods Compared** | **95% Confidence Interval for mean difference** | **p-value for mean difference=0** |
|---|---|---|
| SIROCCO vs. EXTENDED-MG | (-0.003, 0.188) | 0.057 |
| SIROCCO vs. MG | (0.016, 0.149) | 0.015 |
| SIROCCO vs. NON-OP SIROCCO | (0.090, 0.224) | < 0.001 |
| SIROCCO vs. INFORMED-RANDOM | (0.113, 0.266) | < 0.001 |
| SIROCCO vs. RANDOM | (0.211, 0.386) | < 0.001 |

**Table 3**: Inexact Matching Comparisons Using the Nonparametric Bootstrap

Table 2 shows that SIROCCO is clearly superior to all five of the other methods on the exact matching criteria. For each comparison, the interval for mean difference is positive, in favor of SIROCCO, and the probability of SIROCCO having a higher mean is greater than 99%. Taking 95% to be the threshold, as is customary, all five of the findings in this table are statistically significant.

[10] The data was non-Gaussian because the experimental results did not produce anything like a bell-shaped curve. The nonparametric bootstrap method is applicable when observations (i.e., F-Measures) are independent and the likelihood of

Table 3 shows that SIROCCO is superior, in a statistically significant way, to MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM on the inexact matching criteria. The only exception is the comparison of SIROCCO to EXTENDED-MG, for which the probability of a mean difference = 0 is 94.3%, a marginally significant result[11].

This experiment penalized SIROCCO for citing relevant codes and cases that the BER did not cite. While the board is the best "gold standard" available for such an experiment, it is not perfect. One could claim, for instance, that the NSPE BER's opinions do not always cite all of the relevant codes and cases. For example, they may have cited a minimum amount of material required to support their conclusion. Thus, in a supplemental experiment, two ethics graduate students were asked to evaluate the extra code and case citations for the trial cases made by SIROCCO and SIROCCO's closest competitor, EXTENDED-MG. For each additional code and case suggested by the two methods, the evaluators were asked to indicate whether the extra suggested item was reasonable or not.

After verifying that inter-rater reliability was satisfactory, SIROCCO's and EXTENDED-MG's F-Measures for the 58 trial cases were recalculated, counting the extra citations rated as "reasonable" by the evaluators as Board citations. For SIROCCO, the recalculated mean F-Measures were 0.36 for exact matching and 0.58 for inexact matching. For EXTENDED-MG, the recalculated mean F-Measures were 0.25 for exact matching and 0.46 for inexact matching. The nonparametric bootstrap procedure now showed a significant difference between the accuracy of SIROCCO and EXTENDED-MG on both the exact and inexact match criteria. For both criteria, the confidence level of a difference (in favor of SIROCCO) was now at least 99%.

To test whether SIROCCO's temporal knowledge makes a difference in the accuracy of its predictions, the trial cases were processed by an ablated version of the program, NON-TEMP SIROCCO, which did not employ temporal knowledge. NON-TEMP SIROCCO provides the full functionality of SIROCCO with the exception that it doesn't check the consistency of temporal relations across matched cases. As with the initial experiments, the results of NON-TEMP SIROCCO were compared against the suggestions made by the board and the F-Measure calculated for each individual sample and as a mean value over all samples. These results were then compared to the output of the standard version of SIROCCO, which did apply temporal knowledge. The differences between SIROCCO with and without its temporal knowledge were essentially negligible.

---

seeing any particular F-Measure value is the same from observation to observation.

[11] Although EXTENDED-MG was *not* the second best inexact matching method according to the mean F-Measure (see Figure 15), the nonparametric bootstrap procedure is a more precise measure that accounts for variances in the differences between methods. Using this procedure, EXTENDED-MG's results were closer to SIROCCO's than MG's.

# 6 Discussion

The experiments confirmed the hypothesis that SIROCCO's core operationalization techniques help it to make accurate predictions of the principles and past cases that are relevant in the analysis of new cases. The most compelling evidence of this is SIROCCO outperforming NON-OP SIROCCO in the ablation experiment. While both SIROCCO and NON-OP SIROCCO use the same case representation and Stage 1 retrieval method, NON-OP SIROCCO makes no use of the core set of operationalization techniques, most significantly *Principle* and *Case Instantiations*. The instantiations are the primary way in which codes and cases become defined extensionally in a way that SIROCCO can reuse in analyzing new cases.

The effect is substantial. For exact matching, SIROCCO's operationalization information leads to performance that is more than 60% better than NON-OP SIROCCO (.21 to .13). In the inexact match, the improvement is almost 50% (.46 to .31). The core operationalization techniques make a bigger contribution in the exact match test, most likely because inexact matching leaves open more opportunity for "lucky guesses" since a number of codes might match the same abstract code.

The results provide evidence that SIROCCO's use of operationalizations help in identifying cases and codes that are relevant, according to the ethical review board's own standard. In other words, the experiment indicates that the model is accurate and that the modeled operationalizations are a crucial part of the model. Conversely, the results provide evidence that the ethical review board's explanations extensionally define applicability and relevance conditions for the code provisions and past cases. The conceptual links the board draws between critical facts and code provisions/past cases provide information valuable for retrieval in new cases.

Empirical results and conclusions of this type have not, to my knowledge, been demonstrated and supported in any prior research. A key contribution of the work is that an AI model was developed that enabled testing this effect through ablation.

Further evidence of the advantages of operationalization information is the fact that SIROCCO outperformed MG and EXTENDED-MG. This shows that SIROCCO is a more powerful retrieval method than the most likely competitor for this type of task, a full-text retrieval method. The fact that SIROCCO outperformed EXTENDED-MG which, in turn, outperformed MG, is also significant. Since EXTENDED-MG makes at least limited use of operationalization information (i.e., the selection of codes according to frequency of citation) its improvement over MG also supports the hypothesis.

Unfortunately, the results did not show that SIROCCO's temporal knowledge contributed to its accuracy. This was surprising. Intuitively, temporal orderings of events are important in ethical

analysis. For instance, one can be expected to report a dangerous situation only after learning of it, not before. However, the latter case would not appear in the NSPE BER cases because it so obviously does not involve a moral duty. In a case base such as SIROCCO's it might be a rare event that pairs of cases exist such that a difference in temporal ordering leads to different ethical interpretations.

# 7 Comparison to Related Work

This work applies techniques from AI and Law research to a new and in some ways more challenging domain, ethical reasoning.

To be sure, ethical and legal reasoning share much in common. Cases in ethics and law both involve concrete factual scenarios presenting issues for decision. In both domains, the decision makers often make their decisions by applying abstract principles, norms, or laws in specific factual contexts. In each kind of reasoning, operationalization techniques extensionally define the principles by example and flesh out the meanings of the abstract normative standards. In legal opinions, a judge rationalizes his or her decision by explaining how and why the principles, norms and legal rules apply in the problem context. Judges compare problems to past cases or precedents to resolve conflicts among competing principles and to justify conclusions. They also pose hypothetical variations of the scenarios to probe and bolster their analyses, much as ethicists do.

On the other hand, compared to the legal domain, ethics involves a much less-explicit model of argumentation. Ethical arguments are typically more free-form in style and structure. Typically, the decision-making process in engineering ethics is not adversarial. There is no formal "dance" of adversaries presenting arguments, counter-arguments, rebuttals, and surrebuttals, no pleadings and proofs. Past ethics cases may be more-or-less persuasive, but they do not have a precedential effect, as they do in the legal domain. Although an organization such as the NSPE BER provides examples of how engineering ethics problems have been argued and resolved, the Board's opinions are purely educational; they are not binding in the same sense as legal decisions.

Decisions in engineering ethics cases are not constrained to binary conclusions (e.g., plaintiff winning or losing) as is often the case in law[12]. An ethics case may be resolved in multiple ways involving different possible actions and outcomes. For instance, in deciding whether a particular action is ethical or unethical in a case, the NSPE BER often suggests ways in which the protagonist might have avoided the ethical dilemma altogether or ways to correct an unjust action after the fact.

[12] Although most legal cases have binary outcomes, with a winner and a loser, not all do, for instance property distribution and child custody cases.

The goal in evaluating engineering ethics problems is not to assign blame and/or punishment for unethical actions, as often occurs in legal cases; rather, it is to provide an opportunity for interested parties to consider the ethical ramifications of alternative actions and choose the best course. It is important for engineering practitioners to recognize and understand the conflicting values in specific fact situations and to learn to apply "creative middle way" solutions to those situations (i.e., resolutions that at least partially meet each of the conflicting values) [Harris *et al.*, 1999, p. 64-72].

Finally, the two domains differ in terms of access to case data. The legal domain has an extensive body of on-line cases, available through full-text retrieval services such as Westlaw and through a vast array of substantive conceptual indexes. The engineering ethics domain, by contrast, offers much less sophisticated resources with far fewer case examples and opinions and far less conceptual indexing.

SIROCCO's focus on information retrieval is aimed, in part, at exploring how on-line case retrieval for engineering ethics can be improved. Its approach to intelligent retrieval is based on computationally representing the operationalizing links between abstract principles and factual case narratives. To the extent that its computational model succeeds, it could be applied in turn in the legal domain, where the same operationalization techniques apply.

Earlier interpretive CBR programs have employed extensional methods to define statutory legal concepts in terms of cases. CABARET represented tax concepts using dimensions and cases [Rissland and Skalak, 1991]. GREBE represented and applied portions of judges' explanations why statutory predicates in workmen's compensation law were satisfied or not in particular cases [Branting, 1991, 2000].

SIROCCO, however, is the first CBR program to apply extensional methods to the domain of professional ethics. It models how decision makers' explanations of their decisions in ethical cases operationalize abstract ethical principals. It demonstrates how to represent that operationalization information computationally and use to it to improve information retrieval of relevant ethical principles and cases.

Since its approach is closest to GREBE's, a more detailed comparison is in order. GREBE represents legal cases using a relational language and semantic nets; it implements a kind of instantiation to flesh out the meaning of open-textured statutory predicates. SIROCCO, however, improves upon GREBE (and other earlier interpretive CBR efforts) in a number of ways.

First and perhaps most importantly, SIROCCO's extensional model of how abstract principles and past cases accrue meanings through operationalizations is more general than GREBE's use of instantiation. GREBE does not address abstract principles like the ethics code provisions, nor does

it have equivalents to SIROCCO's operationalization techniques for grouping principles or cases, resolving conflicts between principles, instantiating cases, elaborating principles, hypothesizing facts, revising principles, and reusing operationalizations.

Second, SIROCCO represents cases in a more general way as temporally-ordered, narrative descriptions of events. GREBE's case representation focused on representing the events in relation to the court's explanations, while CABARET, BankXX [Rissland *et al.*, 1996] and CATO [Aleven 1997; Ashley and Aleven 1997] describe cases in terms of dimensions or factors. SIROCCO's ETL provides a total of 190 actions and events compared to, for instance, 70 to 90 in GREBE. SIROCCO is also capable of matching these actions and events at various levels of abstractions, something that GREBE did not do. SIROCCO's representation includes formally-defined temporal relations among facts, and a well-defined algorithm for matching temporal relations. None of the other interpretive CBR programs provides such a capability.

Third, SIROCCO has relatively broad domain coverage as compared to other interpretive CBR systems such as HYPO [Ashley 1990] and CATO which handled trade secrets cases exclusively[13]; GREBE which reasoned only about workers' compensation cases; CABARET which processed only home-office tax deduction cases; and BankXX which handled only Chapter 13 personal bankruptcy cases. Although topics in the legal domain and topics in engineering ethics are not directly comparable, it appears that the topical area of each of the earlier systems is narrower than the full set of Selected Topics (see Table 1). If one considers that SIROCCO has also been shown to address cases *outside* the Selected Topics group, it is clear that SIROCCO provides wider domain coverage, at least for the purposes of intelligent retrieval.

Fourth, SIROCCO's case acquisition web site, with its examples, guidelines, ontology and limited language, provides some practical means for achieving consistency of representation across cases. Its generalized matching techniques and inexact matching of actions and events at various levels of abstractions also reduce the need for perfect consistency. Indeed, *twelve* case enterers represented SIROCCO's 242 cases. None of them had been involved in developing the program. Other interpretive CBR programs that use a limited language, such as GREBE and SWALE [Leake, 1991], critically depend on maintaining consistency of representation across cases without providing any practical means to achieve it. For instance, GREBE's structural mappings fail unless cases are consistently represented, a task for the program designer to perform. Of course, web technology was not available when the earlier research occurred, but presumably comparable non-

[13] It may be true that CATO "is not specific to trade secrets law" [1997, p. 41]. This claim might also be made about all of the systems listed here with respect to their specific domains. However, none of these systems actually *demonstrated* wide a domain coverage as SIROCCO.

Internet case acquisition programs could have been developed in an attempt to achieve case consistency.

On the other hand, most of the other interpretive CBR programs generate arguments that are more detailed than SIROCCO's case analyses. For instance, GREBE's representation of a court's precedent-setting explanations in terms of causal and evidential relations enables it to analyze target problems and construct detailed legal arguments. CATO also generates detailed alternative arguments that a partially matched source case is reasonably close (or not) to a target case, by utilizing abstract factors.

Although SIROCCO does not generate detailed arguments, argumentation and decision making in ethics are somewhat less constrained, and thus harder to model, than legal argumentation. Nevertheless, SIROCCO does accommodate information retrieval for ethical decision making and takes advantage of the beneficial effects of operationalization in a way that has never been modeled computationally even in the legal domain. In earlier work, Kevin Ashley and I modeled the generation of more precise and issue-oriented ethics case comparisons in TRUTH-TELLER, a computer program that reasons about reasons in comparing pairs of truth telling dilemmas [McLaren and Ashley, 1995; Ashley and McLaren, 1995]. Such comparisons would be important in generating ethical arguments.

Another advantage of some interpretive CBR programs is their ability to combine different reasoning methods. GREBE and CABARET combine rule-based reasoning (RBR) and case-based reasoning. SIROCCO relies exclusively on cases for retrieving appropriate codes and cases. On the other hand, while SIROCCO does not actually apply rules in its representation and reasoning, it explores an important aspect of the relationship between CBR and RBR: the way in which case explanations extensionally define abstract rules or principles.

One might ask whether deontic logic could be used to model SIROCCO's reasoning. The goal of deontic logic is the creation of a logical formalism for resolving conflicting normative principles. Some research has been aimed at recasting case-based reasoning, like that in the HYPO system, into a deontic logic framework [Horty, 1999]. The objective is to provide a principled way to arbitrate conflicting rules, an effort, which, if successful, would obviate the need for operationalizations in modeling the conflicts among principle. It would be difficult, however, to recast SIROCCO's representation and algorithm in deontic terms. SIROCCO's EETL language represents cases as narratives, fairly rich in detail. It is also not clear how the deontic approach would bridge the gap between specific fact situations and the presumably highly abstract, conflict-resolving logical rules.

There is a cost to using the SIROCCO approach. It takes time and effort to represent a problem situation as a Fact Chronology for input to SIROCCO. A full-text retrieval system, such as the one used in the experiments reported in this paper, does not have this problem as it deals with a new problem simply as text. As demonstrated by the experiments presented in this paper, however, SIROCCO is more accurate than full-text retrieval. In any case, the hope is that ETL, with its use of a *limited* language focusing on the important *verbs* in a domain, may someday support a textual CBR approach to representing cases semi-automatically [see, e.g., Brüninghaus and Ashley, 1999, 2001]. Representing ethics problems in ETL may also have pedagogical advantages, since it leads students to focus closely on what happened and when. These are lines of research still to be explored.

# 8 Conclusion

The computational model described in this paper represents information about how abstract principles have been applied in analyzing past problems and uses it to retrieve information relevant for analyzing new cases. Although abstract principles are rules, they cannot be applied without some way to "bridge the gap" between the rules' abstractions and concrete fact situations. Ill-defined domains like ethics and the law often lack authoritative or readily-available intermediate rules that can be used to bridge this gap. Nevertheless, decision makers apply abstract principles in deciding specific fact situations and often explain and record their reasoning. In fashioning their explanations, they apply various operationalization techniques in order to draw connections among abstract rules and specific scenarios. These linkages tend to define the principles extensionally; they may be discerned, represented, and applied to new fact situations for purposes of improving information retrieval.

An examination of hundreds of cases decided by an engineering society's Board of Ethical Review led to the identification of a set of nine operationalization techniques commonly used by the Board in explaining its decisions. I hypothesized that the Board's operationalization information could be represented and subsequently used to improve automated information retrieval.

The hypothesis was tested with a computational model comprising: (1) an ontology, knowledge representation language, and case acquisition web site for representing the facts and analysis of the ethics cases and (2) the SIROCCO program with its two-stage matching algorithm and its case base of foundational cases. Each case represents the triggering facts that the Board deemed relevant to the application of principles and past cases, groupings of principles and cases, and other operationalization information.

Experiments with SIROCCO confirmed the contributions of a core subset of the operationalization techniques, whose effects could be compared objectively with the board's opinions. The operationalization information allowed the program to more accurately predict principles and past cases that are likely to be relevant in the analysis of new cases than it could without the operational information. This performance difference is achieved by representing and recording the linkages between principles and cases drawn by the Board as it explains its decisions. It is evidence of the way in which the Board extensionally defines principles and cases.

In addition, SIROCCO performed significantly better than several competitor methods, including a full-text retrieval system. In fact, augmenting that information retrieval system with some operationalization information also improved its performance.

The work that is reported in this paper is significant in at least two respects. First, it demonstrates the importance of operationalization as a method for dealing with abstract principles using past case explanations. In particular, the work shows the value of operationalization in an ill-defined domain, engineering ethics, by identifying, cataloguing, implementing, applying, and testing operationalization techniques.

SIROCCO can be thought of as a general model for "operationalized principle and case retrieval" in ill-defined domains where operationalization techniques are in use. Candidate domains need to have, at minimum: (1) a documented set of highly abstract rules (i.e., principles, codes, theories, or policies) and (2) a recorded set of cases in which the abstract rules are applied to reach and rationalize a decision. Each case must include a fact description, at least one question to be decided in the context of the facts described, an outcome, and an analysis of the fact description and explanation of the decision that include citations to the codes, theories, principles, policies, and other cases. One would also need to verify that the operationalization techniques discussed in this paper are used by decision makers in the new domain. Finally, the engineering ethics ontology would need to be extended to include new Fact Primitives, Actors, and Objects relevant to the new domain. To the extent that the new domain deals with engineers' roles and relationships, most of the existing ontology would apply; the matching algorithm would likely apply to any candidate domain satisfying the above requirements.

The work's second major contribution is to demonstrate how AI techniques make possible an empirical investigation of a phenomenon in an ill-defined domain. Indeed, without AI, theorists in the domain probably could not study the phenomenon experimentally. Ethicists would agree that operationalization is important. They speak of a dialectical interaction in which principles inform the decisions of cases and decided cases modify and refine the meanings of principles [Arras, 1994]. Due to the nature of their field of study, however, it is highly unlikely that ethicists could

evaluate this phenomenon empirically without the help of AI techniques. By building a computational model, I was able to test the hypothesis and demonstrate the effects of operationalization objectively. The AI model explicitly represented the ethics case facts and operationalization information in the ethicists' opinions. SIROCCO's code and case suggestions were compared objectively with the NSPE Board's code and case citations for the same cases. Since the operationalization information was represented in a modular way, it could be turned on and off selectively, thus enabling an ablation experiment to focus on the contribution made by that knowledge.

As argued in the previous section, this work extends AI modeling techniques to a domain, ethical reasoning, that is even more ill-defined than the law. The law is, however, another natural domain for applying SIROCCO. The nine operationalization techniques and knowledge representation tools generated in this project may well provide new insights into or techniques for retrieving legal texts. Clearly, the legal domain satisfies the minimum constraints discussed above: abstract principles, published laws, theories, and policies. Judges decide legal cases and publish written opinions in which they explain how and why the abstract rules apply in concrete circumstances. In those opinions, judges appear to use many of the operationalization techniques used by SIROCCO. For instance, there is little doubt that Principle and Case Instantiations are relevant; judges interpret a new case's facts in light of highly abstract rules as applied in precedent cases and vice versa. Representing legal scenarios as narrative lists of steps and annotating the applications of abstract rules with temporally ordered clusters of questions presented and relevant facts may provide new leverage for intelligently retrieving abstract rules and past precedents.

Since legal domains usually involve both legal and ethical analyses, a particularly interesting generalization of SIROCCO would be to explore its use in representing and integrating both the legal *and* ethical aspects of a domain. For instance, disclosing a client's confidential information raises questions of trade secret law and contract, as well as ethical questions. Much of SIROCCO's knowledge representation applies to trade secret scenarios arising from engineers' disclosure of confidences. In addition, there is a well-developed AI and Law methodology for representing trade secret law. The points at which legal and the ethical analyses lead to divergent results elucidate the true meaning of abstract legal and ethical rules. Generally, such points can be illustrated best (and, perhaps, exclusively) by concrete case examples, an interesting ramification of the way in which such abstract normative rules are defined extensionally.

Other extensions to the SIROCCO model have been explored and/or developed. In particular, experiments with how the program can "know" the limitations of its expertise have been executed [McLaren and Ashley, 2001]. We hypothesized that it would sometimes be better for the program

to admit that it lacks the knowledge to suggest relevant codes and past source cases to an input target case. We identified and encoded strategic meta-rules to help it decide. The meta-rules are sensitive to the varying levels of analysis performed by SIROCCO. For instance, one meta-rule measures the quality of SIROCCO's surface-level matches (i.e., Stage 1) while another measures the quality of the program's structural matches (i.e., Stage 2). If the rules indicate that the quality of match is poor for a particular target case, the program does not provide suggestions for that case. Experiments demonstrated that the meta-rules improve the program's overall advice-giving performance.

Another extension of the research is to incorporate SIROCCO into an intelligent tutoring system for engineering ethics. As an intelligent retrieval component of a tutoring system, SIROCCO could help engineers and students bridge the gap between cases and abstract principles [Goldin, Ashley, and Pinkus, 2001]. Pedagogically, students may benefit from representing problem scenarios explicitly as narratives of temporally-ordered events (e.g., it induces students to consider more carefully the facts of a case.) Using the program, students could retrieve relevant examples illustrating applications of the principles. Since full-text retrieval schemes alone cannot generate explanations, SIROCCO's explanations of its outputs may be very useful in a tutoring context.

# 9 Acknowledgements

This paper reports on work performed in partial fulfillment of my Ph.D. dissertation at the University of Pittsburgh in the Intelligent Systems Graduate Program. I would like to thank the members of my dissertation committee: Dr. Martha E. Pollack, Dr. Manuela M. Veloso, Dr. Bruce G. Buchanan, and especially Dr. Kevin Ashley who served as my committee chair and intellectual mentor.

I would also like to thank the anonymous reviewers of the paper and the editors of this special edition of the AI Journal, who provided many helpful suggestions and comments.

# 10 References

Aleven, V. A. 1997. *Teaching Case-Based Argumentation Through a Model and Examples*. Ph.D. Dissertation, University of Pittsburgh.

Allen, J. F. 1983. Maintaining Knowledge about Temporal Intervals. *Communications of ACM* 26(11), 832-843.

Arras, J. D. 1994. *Principles and Particularity: The Role of Cases in Bioethics* 69 Indiana L. J. 983, 1006.

Ashley, K. D. and Aleven, V. A. 1997. Reasoning Symbolically About Partially Matched Cases. In the *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence* (IJCAI-97). Nagoya, Japan. 335-341.

Ashley, K. D. and McLaren, B. M. 1995. Reasoning with Reasons in Case-Based Comparisons. In the *Proceedings of the First International Conference on Case-Based Reasoning*, Sesimbra, Portugal.

Ashley, K.D. 1990. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals.* Cambridge: MIT Press.

Berman, D. H. and Hafner, C. D. 1986. Obstacles to the Development of Logic-Based Models of Legal Reasoning. In *Computer Power and Legal Language*. C. Walter, editor, 185-214, Greenwood Press.

Bonissone, P. and Ayub, S. 1992. Representing Cases and Rules in Plausible Reasoning Systems. In the *Proceedings of the 1992 IEEE International Conference on Tools with AI*. Arlington, VA.

Branting, L. K. 2000. *Reasoning with Rules and Precedents.* Kluwer: Dordrecht, The Netherlands

Branting, L. K. 1991. Building Explanations from Rules and Structured Cases. *International Journal of Man-Machine Studies*, 34 (6): 797-837.

Brüninghaus, S. and Ashley, K. D. 2001. The Role of Information Extraction for Textual CBR. In: *Proceedings of the 4th International Conference on Case-Based Reasoning*, Vancouver, Canada. 74-89.

Brüninghaus, S. and Ashley, K. D. 1999. Bootstrapping Case Base Development with Annotated Case Summaries (Althoff, K.-D., *et al*. ed.) *CBR Research and Development* (ICCBR-99) Lecture Notes in AI No. 1650, 59-73. Springer: Berlin.

Davison, A. C. and Hinkley, D. V. 1997. *Bootstrap Methods and Their Application*. Cambridge U. Press.

Forbus, K. D., Gentner D., and Law, K. 1994. MAC/FAC: A Model of Similarity-based Retrieval. *Cognitive Science* 19, 141-205.

Gardner, A. 1987. *An Artificial Intelligence Approach to Legal Reasoning*. Cambridge, MA: MIT Press.

Gentner, D. 1983. Structure-mapping: A Theoretical Framework for Analogy. *Cognitive Science*, 7, 155-170.

Ginsberg, M. 1993. *Essentials of Artificial Intelligence*. San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Goldin, I. M., Ashley, K.D., and Pinkus, R. L. 2001. Introducing PETE: Computer Support for Teaching Ethics. In the *Proceedings of the Eighth International Conference on Artificial Intelligence and Law* (ICAIL-2001).

Harris, C. E., Pritchard, M. S., and Rabins, M. J. 1999. *Engineering Ethics: Concepts and Cases*. 2nd edition Belmont, CA: Wadsworth.

Horty. J., 1999. Precedent, Deontic Logic, and Inheritance. In the *Proceedings of the Seventh International Conference on Artificial Intelligence and Law* (ICAIL-99).

Jonsen, A.R. and Toulmin, S. 1988. *The Abuse of Casuistry: A History of Moral Reasoning*. Berkeley: U. California Press.

Kettler, B. P., Hendler, J. A., Andersen, W. A., and Evett, M. P. 1994. Massively Parallel Support for Case-Based Planning. *IEEE Expert*, February 1994, 8-14.

Kolodner, J. 1993. *Case-Based Reasoning*. San Mateo: Morgan Kaufmann.

Koomen, J. A. 1989. *The TIMELOGIC Temporal Reasoning System*. Tech. Rep. 231, Computer Science Dept., U. Rochester, NY.

Leake, D. B. 1991. An Indexing Vocabulary for Case-Based Explanation. In the *Proceedings of AAAI-91*. 10-15.

Lewis, D. D., Schapire, R. E., Callan, J. P., and Papka, R. 1996. Training Algorithms for Linear Text Classifiers. In the *Proceedings of the 19th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*. Zurich.

McLaren, B. M. and Ashley, K. D. 2001. Helping a CBR Program Know What it Knows. In the *Proceedings of the 4th International Conference on Case-Based Reasoning*, Vancouver, Canada. 377-391.

McLaren, B. M. and Ashley, K. D. 2000. Assessing Relevance With Extensionally Defined Principles and Cases. In the *Proceedings of the 17th National Conference on Artificial Intelligence*, 316-322.

McLaren, B.M. 1999. *Assessing the Relevance of Cases and Principles Using Operationalization Techniques*. Ph.D. Dissertation, University of Pittsburgh.

McLaren, B. M. and Ashley, K. D. 1999. Case Representation, Acquisition, and Retrieval in SIROCCO (Althoff, K.-D., *et al*. ed.) *CBR Research and Development* (ICCBR-99). Lecture Notes in AI No. 1650. 248-262. Springer: Berlin.

McLaren, B. M. and Ashley, K. D. 1995. Case-Based Comparative Evaluation in TRUTH-TELLER. In the *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*. Pittsburgh, PA.

Mostow, J. 1983. Machine transformation of advice into a heuristic search procedure. *Machine Learning*, vol. 1.

National Society of Professional Engineers (NSPE) 1958-1998. Opinions of the Board of Ethical Review, Volumes I through VIII. Alexandria, VA: the National Society of Professional Engineers.

National Society of Professional Engineers (NSPE) 1996. *The NSPE Ethics Reference Guide*. Alexandria, VA: the National Society of Professional Engineers.

Pinkus, R., Ashley, K., Chi, M. and Moore, J. 1997. "Modeling Learning to Reason with Cases in Engineering Ethics: A Test Domain for Intelligent Assistance." NSF Proposal No. 9720341. Project Statement.

Resnick, R. and Halliday, D. 1967. *Physics.* New York: John Wiley & Sons, Inc.

Rissland, E. L., Skalak, D. B., and Friedman, M. T. 1997. Evaluating a Legal Argument Program: The BankXX Experiments. *Artificial Intelligence and Law* Volume 5. The Netherlands: Kluwer Academic Publishers.

Rissland, E. L., Skalak, D. B., and Friedman, M. T. 1996. BankXX: Supporting Legal Arguments through Heuristic Retrieval. *Artificial Intelligence and Law* 4:1-71.

Rissland, E. L. and Skalak, D. B. 1991. CABARET: Statutory Interpretation in a Hybrid Architecture. In *International Journal of Man-Machine Studies*, 34 (6): 839-887.

Rissland, E. L. 1978. *The Structure of Mathematical Knowledge*. A.I. Technical Report No. 472. Massachusetts Institute of Technology.

Thagard, P., Holyoak, K.J., Nelson, G. and Gochfeld, D. 1990. *Analog Retrieval by Constraint Satisfaction*. Technical Report CSL-Report 41, Princeton U.

Toulmin, S. E. 1958. *The Uses of Argument*. Cambridge, England: Cambridge University Press.

Twining, W. and Miers, D. 1976. *How to Do Things With Rules*. London: Cox and Wyman, Ltd.

van Rijsbergen, C. J. 1979. *Information Retrieval*. London: Butterworths, Second edition.

Witten, I., Moffat, A., and Bell, T. C. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. San Francisco, CA: Morgan Kaufmann, Second edition.