**Submitted to the *AI and Law Journal***

# BankXX: Supporting Legal Arguments through Heuristic Retrieval

Edwina L. Rissland
David B. Skalak
M. Timur Friedman

Department of Computer Science
University of Massachusetts
Amherst, MA 01003

rissland@cs.umass.edu

## Abstract

The BankXX system models the process of perusing and gathering information for argument as a heuristic best-first search for relevant cases, theories, and other domain-specific information. As BankXX searches its heterogeneous and highly interconnected network of domain knowledge, information is incrementally analyzed and amalgamated into a dozen desirable ingredients for argument (called *argument pieces*), such as citations to cases, applications of legal theories, and references to prototypical factual scenarios. At the conclusion of the search, BankXX outputs the set of argument pieces filled with harvested material relevant to the input problem situation.

This research explores the appropriateness of the search paradigm as a framework for harvesting and mining information needed to make legal arguments. In this first of two articles, we describe how legal research fits the heuristic search framework and detail how this model is used in BankXX. We describe the BankXX program with emphasis on its representation of legal knowledge and legal argument. We describe the heuristic search mechanism and evaluation functions that drive the program. We give an extended example of the processing of BankXX on the facts of an actual legal case in BankXX's application domain—the good faith question of Chapter 13 personal bankruptcy law. We discuss closely related research on legal knowledge representation and retrieval and the use of search for case retrieval or tasks related to argument creation. Finally we review what we believe are the contributions of this research to the understanding of the diverse disciplines it addresses.

**This research supported in part by grant No. 90-0359 from the Air Force Office of Sponsored Research and NSF grant No. EEC-9209623 State/University/Industry Cooperative Research on Intelligent Information Retrieval.**

## Contents

## Part I: The Approach

### 1. Introduction

In this article we present our research on the problem of perusing and gathering information for use in legal argument. In particular, we discuss our program BankXX and its use of the heuristic search paradigm as a computational framework for this information harvesting task.

This research attempts to bring together a number of ideas about artificial intelligence and about law. Its ideas unite information retrieval, architecture and control of AI programs, search, case-based reasoning, legal research, legal knowledge representation and indexing, and legal argument.

Some of the points we will touch upon in our description of the BankXX system are these:

- BankXX is rooted in the task of performing legal research and provides a framework for modeling research strategies.
- The process of gathering information for legal argument can be usefully framed as classic heuristic best-first search.
- The presence of multiple types of legal knowledge and multiple ways to view and index it can be used to advantage in our task.
- Retrieval of cases and other legal knowledge can fruitfully use a combination of knowledge-based indexing and heuristic search.
- Aspects of legal retrieval for argument generation can be modeled by a computer program that relies on a search-driven control strategy.
- *Argument pieces* can be used to represent argument and define an evaluation function.

These ideas are not all new. For example, the body of research on conceptual legal retrieval, for example [Hafner 1987a, 1987b; Bing 1987] has proposed the organization of legal knowledge as a semantic network that implicitly permits multiple indexing. Work in cased-based reasoning has also made use of multiple indexing [Kolodner, 1983; Turner, 1988]. But this work brings together both these old and a number of new ideas into a single framework.

This work complements and extends our own work on legal argument. For instance, because of its more bottom up nature, this work on BankXX complements our past work on top-down control of legal reasoning, for instance, through argument strategies and tactics, as in CABARET [Rissland & Skalak, 1991; Skalak & Rissland, 1992] and context-sensitive skeletal task plans, as in FRANK [Rissland et al., 1993]. It broadens the scope of what type of knowledge has been explicitly represented in our systems, for instance, legal theories and prototypical factual stories. By explicitly representing legal theories (in terms of domain factors), it extends our earlier work on HYPO [Rissland et al., 1984; Ashley, 1990]. It extends the use of "dimensions" to the "meta" realm of argument assessment by using *argument factors* to evaluate and compare arguments. It complements our purely precedent-based representation of argument (e.g., "3-ply arguments") by inclusion of a more diverse set of components—called *argument pieces*—in BankXX's representation of argument (e.g., leading cases, applicable legal theories).[1]

This research also has a strong evaluation component, and we have striven to understand how our program performs, compared to previous programs, compared to legal opinions, and relative to different parameter settings within the program itself. Consistent with the trend towards more evaluation of AI research, we have performed an extensive series of experiments. These evaluative aspects of our project are reported on in the companion article.

This article first introduces the BankXX system generally by discussing in turn each of the bulleted points we have made. It then describes the domain of the BankXX program, an aspect of U.S. federal bankruptcy law concerning personal bankruptcies under Chapter 13. The program itself is described, with emphasis on its representation of its particular area of legal knowledge and its representation of argument. Then we detail the mechanisms of heuristic search, such as the evaluation functions, that drive the program. Once all these pieces are in place, we give an extended example of the processing of BankXX on the facts of an actual legal case. We conclude this articles with a discussion of closely related research on legal knowledge representation and retrieval and the use of search for case retrieval or tasks related to argument creation. Finally we review what we believe are the contributions of this research to the understanding of the diverse disciplines it addresses.

## 1.1. Legal Research Strategies

The approach to information gathering in BankXX[2] is similar to what a junior associate in a large law firm might do when charged with the task of providing information to support an argument that is being crafted by a senior attorney. Using indices and connections provided by legal materials, the junior lawyer must search through volumes of primary sources (e.g., opinions, statutes) and secondary legal commentary (e.g., treatises, law journal articles) for the legal cases, legal theories, and statutory and regulatory citations to underpin an argument on a designated issue. Additionally, his[3] search must be completed within a certain time frame and is further constrained by the resources, such as legal materials, that are actually available and the intended use of the research (e.g., internal memorandum, formal brief) [Rissland, et al., 1993]. Obviously, exhaustive blind search is not viable because of the sheer volume of legal materials available. Thus, the junior associate must use heuristics to manage his research activities: researching new material

[1]Or course, there is much more that could be included, such as jurisdictional or procedural aspects, both of which are important [Berman & Hafner, 1991].

[2]We pronounce the name of this program as "Bank-ex-ex".

[3]Masculine pronouns should be read to encompass both males and females.

based on approximate, though usually accurate, ideas of what's important to an argument, both in its details and in its overall quality. That is, the process of gathering for argument fits the classic model of heuristically-guided best first search. BankXX reifies that model.

In researching a legal issue, it is often the case that a lawyer has a very good general sense of what types of information are needed to mount a convincing argument, even if he does not actually know the specifics of the legal area, such as particular precedents. In addition, a researcher knows how to exploit this general knowledge in the context of his specific problem to find more knowledge. General knowledge used in search includes:

1. what types of domain knowledge exist (e.g., cases, treatises, annotated statutes, legal theories) and how they are interconnected (e.g., cases cite other cases, cases can announce legal theories, cases invoke legal theories);
2. what basic pieces of information are needed to make an argument (e.g., good supporting cases, cases that trump the opponent's cases, a viable legal theory, an appealing story to tell) and how these support each other (e.g., supporting cases give rise to justifying analogies, a prototypical story can help frame an issue);
3. what makes an argument a good one (e.g., to the extent that one uses central cases an argument is better than one that uses rarely cited outliers; to the extent that supporting cases fit under one theory, an argument is better than one where a variety of theories must be cobbled together).

These general notions about legal knowledge and legal argument help drive the search for specific information to flesh out an argument. At each step of research, such general research knowledge plus the emerging problem-related knowledge already discovered provides a scaffold to help frame and mount new knowledge-harvesting forays. In BankXX such general knowledge is used to define certain computational mechanisms—evaluation functions, argument pieces, etc.— needed to carry out heuristic search and represent and complete the task of information harvesting for legal argument.

## 1.2. Argument Generation as Search

In our view the generation of argument can be viewed as heuristic search. At each stage in developing an argument, choices need to be made. Should one seek a broad set of supporting cases, anticipate the best cases for the opposing side, or create a telling hypothetical? Each choice takes the emerging argument to a new state of development. Limited resources force the arguer to make choices about which avenues to pursue.

In one implementation of argument as search, the search space would be the space of all arguments, the start state would be an empty argument, and the search operators would represent ways to advance the argument. However, our current system models argument as the emerging by-product of the search and research that an expert might perform in a space of domain knowledge. We perform search in the domain space rather than in an argument space in part because we are interested in modeling the legal research activities of attorneys and partly because the indexing fabric of the domain space is better understood. In our approach, domain space search identifies nodes that contain domain knowledge that can support an argument and collects the support in an argument data structure.

BankXX carries out its search in a network of frames representing items of traditional legal materials: legal cases, legal theories, etc. BankXX searches through these using a variation on the classic method of heuristic best-first search:

> BankXX "expands" the "current node" (the material currently being examined) to generate "successor" nodes (new materials to look at); these are placed on a list of "open" nodes (a list of materials to examine); a heuristic evaluation function is applied to them;[4] and the best becomes the new current node, which is then mined for information it can yield for the evolving argument. And the process repeats.

Successor nodes are found by calculating and chasing interconnections through the network used to represent legal knowledge. A heuristic evaluation function assesses a new (opened) node's potential for contributing information to the emerging argument. In the BankXX project, we have used three different evaluation functions. They capture knowledge at one of three levels of abstraction: domain knowledge, knowledge of what pieces of information are needed for an argument, and knowledge about what makes an argument good.

In summary, the BankXX system models the process of creating an argument through legal research as a heuristic search for relevant cases, theories, and other basic information. The information is incrementally analyzed and amalgamated into standard, desirable ingredients for an argument, such as citations to cases, applications of legal theories, and references to prototypical factual scenarios. The research reported here is an experiment to test the appropriateness of the search paradigm as a framework for researching—that is, for retrieving and mining—the knowledge needed to make legal arguments.

### 1.3. Representation of Legal Knowledge

Legal experts working in Anglo-American jurisdictions can access a great variety of information in the course of researching a legal issue. Each of these provides an entry point, or index, into a large library of legal materials. These include:

(1) <u>Traditional citation linkages between cases.</u> A case cites precedent cases for support of the various legal propositions it advances. Indexing services such as *Shepard's Federal Citations* track the citations among many published cases. Citation signals like *see, but see,* and *cf.* reference precedents in precise ways specified in legal style manuals such as *The Blue Book* [BlueBook, 1986]. See [Ashley & Rissland, 1987].

[4]In the BankXX implementation, each node on the OPEN list is (re-)evaluated each cycle. This is a departure from the traditional algorithm. Details are discussed below in Section 3.1.

(2) Legal factors or "dimensions." Many legal areas make use of factors to help frame approaches to resolving legal questions; some come directly from statutes, others arise in the common law. In domains where cases can be compared with respect to a stable set of discernible factors, the factors can be conceptualized and implemented as "dimensions" [Rissland et al., 1984; Ashley, 1990]. Dimensions may be used to index and retrieve cases from a case base and to order precedents by their relevance to a problem situation as in the HYPO and CABARET systems [Ashley, 1990; Rissland & Skalak, 1991].

(3) Legal theories. Courts and advocates usually strive to provide a legal theory as to why a case should be decided a certain way. Legal theories are often explicitly couched in terms of factors or other features to be considered. Knowing what cases have been argued under a theory is a means to access other cases, such as cases in which a theory was clearly held to control a decision. In addition, relations between legal theories themselves, such as refinement, permit "nearby" theories to be retrieved, along with the cases that apply them.

(4) Recurring prototypical fact patterns or legal stories. Generic cases or recurring fact patterns—what could be called *legal stories* or scripts—have been used in legal reasoning [Gardner, 1987] as well as in other domains [Schank, 1990]. If, for example, the current problem involves a former student with large educational debts who files for bankruptcy immediately after graduating, a bankruptcy expert may recall previous student loan cases. In particular, knowing that a legal theory has successfully been applied to cases fitting a particular fact pattern provides a basis for creating analogies to justify applying that theory. Sullivan et al.'s very thorough analysis of bankruptcy law [Sullivan, *et al.*, 1989] uses such *story prototypes* to organize the data and conclusions.[5] For example, factual prototypes provide the titles for six chapters in the book, referring to prototypical debtors such as entrepreneurs, homeowners, women, medical debtors, credit card junkies and repeat bankruptcy filers.

(5) Family resemblance or prototypicality indices. Within a given family of stories, one often has a notion of what is a prototypical or most familiar case. This is true not only in law but across the full range of human experience and has been addressed in cognitive psychology. In particular, measures of family resemblance and prototypicality originating in part from the psychological research of Rosch can be used to create a graded landscape of cases in which the highest peaks or most centrally located instances represent cases with the greatest family resemblance rating [Rosch & Mervis, 1975; see also McCarty, 1983; Bareiss, 1989]. Given a case family, one can find member cases of varying degrees of prototypicality. Given an individual case, one can assess the strength of its membership within a family.

[5]For instance, the prototypical credit card junkie only accounts for less than 2% of total bankruptcies in their study. Medical calamity debtors account for 1% to 2% of bankruptcy debtors. [Sullivan, et al., 1989, p.168, p.188].

(6) Rules. There is no shortage of rules in the law [Twining & Miers, 1982]: statutes; agency regulations; "blackletter" rules (generalizations of case law found in restatements of the law) and the rules of a case (stating the holding of the case). Each type of rule provides a means to access cases: the cases leading to the rule, the cases elaborating the rule, those following the rule, those representing exceptions to the rule, etc. In our CABARET system, such relations between cases and rules were used to define strategies and tactics that ultimately specified what type of case needed to be indexed in a given argumentative task [Rissland & Skalak, 1991; Skalak & Rissland, 1992].

(7) Domain taxonomies. Commercial publishers have also developed indexing schemes, such as the key number system used in WestLaw [West, 1992], in which legal topics are assigned key numbers. Such schemes provide a useful taxonomy of the law and index legal opinions by the topics they address.

(8) Terms of art. Through dictionary, digest, and encyclopedia entries legal practitioners find cases that define, interpret, elaborate and refine the meaning of legal terms whose scope is often the source of litigation (cf. "dictionary-based reminding" [Schank, 1982]).

(9) Other secondary sources. Law review articles and notes, practice manuals (e.g., BNA Tax Management Portfolio [Knobbe, 1986], treatises and other reference works (e.g., Words and Phrases [1994]) organize the domain knowledge and provide links to related legal sources. The volume and variety of the secondary authorities makes for a research task in itself. Kunz and colleagues [1992] describe an interesting experiment in which each of the authors performed a research task and maintained a protocol of the materials consulted; each of this book's four authors of this book on legal research strategies began with secondary authorities. It was noted that secondary authorities provided "insight into pertinent legal theories" [p. 468] as well as citations to primary authority.

Each of these types of legal materials has its own emphasis or imparts its own perspective on legal knowledge. Each displays its own strengths and weaknesses as an indexing medium. While our project deals with primary legal authorities—in particular, knowledge of types (1) through (5)—one should not overlook the role of secondary authority in organizing the research task. For instance, American Law Reports (ALR) is a very useful place to find cases to start one's search. In the next sections we discuss these five types of knowledge and indices in further detail and show how they may be used in conjunction with each other to improve retrieval.[6]

Because legal materials can be viewed from various perspectives and can be linked in various ways, legal knowledge is naturally represented as a highly interconnected

[6] We have previously addressed the role of rules as indexes to legal cases in [Rissland & Skalak, 1991] and [Skalak & Rissland, 1992].

network whose nodes represent various items of knowledge and whose links represent their interconnections. Said another way, the representation of legal knowledge is a graph of heterogeneous nodes rather than a tree (e.g., a discrimination tree) since in a graph, there can be multiple routes to an individual node whereas in a tree, there can be only one.

This provides flexible ways of indexing. This flexibility aids retrieval of information, like cases, in several ways. Multiple paths to cases, found through the sequential application of distinct types of indices, can be coupled with case representations at different abstraction levels, and can yield a finer retrieval granularity. The use of multiple types of indices also increases the robustness of case retrieval in "real world" domains in which noisy cases can be indexed incorrectly. Mis-indexing a case by one index does not render it inaccessible when other indices still provide a path. Finally, from a cognitive vantage point, in a richly connected domain like the law, people use a variety of indices for reminding or for accessing information [Schank, 1982; Rissland, 1978].

However, this rich indexing fabric means there are many choices of how and what information to index, and in search terms, how to wend one's way through a highly branched network. In the face of limited resources (e.g., time), this means that there is a premium placed on effective exploration of the network. In other words, some intelligence is needed to search and harvest information from this rich domain network; an exhaustive search is not practical.[7] In BankXX, knowledge about argument and legal materials—captured, for instance, in the evaluation functions—is used to constrain and guide the search.

BankXX shows how different access paths to information can be found by applying related types of indices in complementary ways and that composite indexing strategies can lead to improved case retrieval.

### 1.4. Indexing and Search

Case-based reasoning (CBR) is usually used in service of a task such as planning, design, argumentation, or teaching; its cases and indices are usually encoded at the domain level [Ashley & Aleven, 1991; Hammond, 1989; Sycara & Navinchandra, 1991; Rissland & Skalak, 1991]. One problem in using CBR is that the indexing may not provide adequate retrieval if the constraints stemming from the task cannot be readily translated into the available indices. This difficulty exists in generating arguments: the vocabulary of the constraints on an emerging argument is different from the indexing vocabulary available for case retrieval. To take an extreme (but real) example, suppose that the cases are full-text legal opinions and Boolean combinations of keywords are the only indices available; further suppose the

[7]This is especially so if one considers the BankXX approach fielded in a realistic library situation, whose materials constitute a network with gigabytes upon gigabytes of information.

requirement of the argument is to supply a case that uses the opponent's best theory, so that one can distinguish the case from the current problem and thereby discredit application of the opponent's theory. The constraints of this task cannot be readily expressed in terms of the available indices: there is a mismatch between the indexing and the task vocabularies.[8] That may be true even in more sophisticated indexing regimes, such as those based on "dimensions" used in our own work, although the problem is not as extreme, since dimensions are designed to captured factors that are important to the task of arguing a claim. In such a situation, information needed for the argumentation task cannot be found by indexing alone. Barring revision of the indexing vocabulary or a re-conceptualization of the domain, some search of the information resources is probably needed.

Indexing and search present two extremes for retrieval. At one extreme, a set of indices may function as database retrieval keys, and no search of the case memory need be done, only whatever minimal search is required to match the database key; cases are pre-indexed to permit immediate retrievals. At the other extreme, search is relied on entirely. Through an evaluation function, spreading activation, planning, blind rummaging, or some other technique, the case space is searched for the desired cases. Search may be needed even in a supposedly well-indexed case base if static indices cannot function as database keys; perhaps the domain is rapidly changing or cases need to be retrieved in ways not anticipated or enabled by the original indexing. We see both indexing and search as useful, and the question is how to combine them.

BankXX bridges the gap between what's available from the indexing schemes and what's needed for the task of argument through best-first search guided by evaluation functions defined at various levels of abstraction. At the lowest level—the domain level—the evaluation function uses only information readily available from indexing at the domain level. At the highest level—the overall argument level—the evaluation uses information addressing the overall substance and quality of the argument. At the intermediate level—the argument-piece level—the evaluation function uses information computable from the domain level but geared to the needs of the argument level.

In summary, BankXX incorporates a hybrid search-indexing approach that couples indexing with exploration of cases and other domain knowledge through best-first search in order to (1) address shortcomings in available indexing structures and (2) increase the leverage obtainable from the existing indices.

[8]An analogous vocabulary gap between instances and their generalizations has been noted by [Porter, Bareiss & Holte, 1990].

### 1.5. Search-driven Control Architecture

The perspective we take in BankXX complements previous work in which we sought to recognize and apply structures of legal argument imposed from the top down [Skalak & Rissland, 1992]. In a complete picture, we believe that argument generation includes a flexible control strategy, combining top-down, bottom-up and island-driving [Erman et al., 1980] strategies. Legal research seems clearly an opportunistic process [Kunz, et al. 1992].

BankXX uses a more bottom-up approach, in which gathering the support for arguments is data-driven, that is, driven by the research materials that are encountered. However, we should note that the task (i.e., filling in argument pieces) and some details of the search model (i.e., evaluation functions) import some top-down concerns into this bottom-up approach.

## 2. The Bankruptcy "Good Faith" Domain

BankXX is instantiated in the area of bankruptcy law for individuals that is covered by Chapter 13 of United States bankruptcy law (11 U.S.C. §§ 1301-1330). Chapter 13 provides a means for individual debtors to obtain relief from debts while keeping much of their property. Under Chapter 13 a debtor pays his creditors according to a court-approved plan that allocates 100% of his disposable income for a period of three to five years. Successful completion of the plan discharges the entire debt, regardless of the portion that is actually repaid. By contrast, Chapter 7 (11 U.S.C. §§ 701-766) is based on liquidation of a debtor's assets to satisfy debts.

There is potential for abuse of the debt-absolving power of Chapter 13. For example, a consumer could take out a large loan and spend the money with no intention of repaying it; a student could take out an educational loan and default on it without even trying to repay the loan. By declaring bankruptcy such a debtor would hope to get away with just repaying a small fraction of what is owed. One way the law is designed to prevent this and other abuse is by requiring that a repayment plan be "proposed in good faith" as required in § 1325(a)(3):

**§ 1325. Confirmation of plan**

(a) Except as provided in subsection (b), the court shall confirm a plan if —

(1) the plan complies with the provisions of this chapter and with the other applicable provisions of this title;

(2) any fee, charge, or amount required under chapter 123 of title 28, or by the plan, to be paid before confirmation, has been paid;

*(3) the plan has been proposed in good faith and not by any means forbidden by law;*

(4) the value, as of the effective date of the plan, of property to be distributed under the plan on account of each allowed unsecured claim is not less than the amount that would be paid on such claims if the estate of the debtor were liquidated under chapter 7 of this title on such date;

(5) with respect to each allowed secured claim provided for by the plan....

(6) the debtor will be able to make all payments under the plan and to comply with the plan.

11 U.S.C. § 1325(a), emphasis added

Since Chapter 13 took effect as part of the Bankruptcy Reform Act in October, 1979, many cases have been litigated around the good faith issue. Evolving case law has elaborated what constitutes "good faith," a term left undefined in the original text of the law. Courts of Appeal for most of the federal circuits have articulated legal theories on the issue; to date the Supreme Court has not. The general approach taken by most courts has been to list a number of "factors" that a bankruptcy court should consider in making its decision. For example, one influential standard was articulated by the Eighth Circuit Court of Appeals in 1982 in *In re Estus*:

"We make no attempt to enumerate all relevant considerations since the factors and the weight they are to be given will vary with the facts and circumstances of the case. However, in addition to the percentage of repayment to unsecured creditors, some of the factors that a court may find meaningful in making its determination of good faith are:

(1) the amount of the proposed payments and the amount of the debtor's surplus;
(2) the debtor's employment history, ability to earn and likelihood of future increases in income;
(3) the probable or expected duration of the plan;
(4) the accuracy of the plan's statements of the debts, expenses and percentage repayment of unsecured debt and whether any inaccuracies are an attempt to mislead the court;
(5) the extent of preferential treatment between classes of creditors;
(6) the extent to which secured claims are modified;
(7) the type of debt sought to be discharged and whether any such debt is nondischargeable in Chapter 7;
(8) the existence of special circumstances such as inordinate medical expenses;
(9) the frequency with which the debtor has sought relief under the Bankruptcy Reform Act;
(10) the motivation and sincerity of the debtor in seeking Chapter 13 relief;
and
(11) the burden which the plan's administration would place upon the trustee."

*In re Estus*, 695 F.2d 311, 317 (8th Cir. 1982)

Note that the *Estus* court leaves open the questions of whether these are all the factors that a bankruptcy court should consider and how are they to be applied: "We make no attempt to enumerate all relevant considerations since the factors and the weight they are to be given will vary with the facts and circumstances of the case." (*id.)* However, *Estus* does give special emphasis to one factor: the percentage of debt repaid to unsecured creditors, stating that "[a] low percentage proposal should cause the courts to look askance at the plan" (*id.*).

There are many "legal theories" in the corpus of cases addressing the "good faith" issue. Examples include a per se minimum payment requirement, which required that a threshold percentage of debts be repaid under a plan (see, e.g., *In re Burrell*, 2 B.R. 650 (1980), reversed, 25 B.R. 717 (1982)), blanket tests of good faith such as "all the facts and circumstances" (see *Barnes v. Whelan*, 689 F.2d 1983 (D.C. Cir 1982)); and a theory consisting of 16 factors borrowed from other cases (*In re Easley*, 72 B.R. 948 (Bkrptcy M.D. Tenn. 1987)). Often the theories can be viewed as related or derived from each other, for instance, *Easley* expands the set of *Estus* theory factors, which it then includes as a subset. One theory might modify individual factors from another theory, or a theory might alter another theory's set of factors and their relative weightings. *Iacovoni* uses a factor-by-factor approach, citing a number of previous applied factors, but adds the superstructure that a set of factors are to be considered to determine whether a good faith effort to make a “meaningful repayment” has been made (*In re Iacovoni*, 2 B.R. 256 (1980)).

The theories have also been subject to changes in the bankruptcy code itself. The Bankruptcy Amendments and Federal Judgeship Act of 1984 added in section 1325(b) a requirement that 100% of a debtor’s disposable income be used in the plan, eliminating the relevance of *Estus* factor (1) in subsequent cases.[9]

[9]The current statute requires in 11 U.S.C. 1325(b)(1) “If the trustee or the holder of an allowed unsecured claim objects to the confirmation of the plan, then the court may not approve the plan unless, as of the effective date of the plan — (A) the value of the property to be distributed under the plan on account of such claim is not less than the amount of such claim; or (B) the plan provides that all of the debtor’s projected disposable income to be received in the three-year period beginning on the date that the first payment is due under the plan will be applied to make payments under the plan.” Subsection (b) was added by the Bankruptcy Amendments and Federal Judgeship Act of 1984.

**Part II: Implementation**

## 3. BankXX System Description

In this section we describe BankXX in some detail. First, we describe the overall architecture of the system. Next we describe the static aspects of the system, including the representation the program uses for its knowledge about argument, cases and other knowledge of the bankruptcy domain. Last, we describe the dynamic workings of the system, including the evaluation functions that drive the heuristic search of the static data structures.

### 3.1. System Architecture

The goal of BankXX is to examine the nodes of a network of domain knowledge in order to harvest information that can be used to support a legal argument. This examination is performed using mechanisms of classical heuristic search. BankXX's control flow is shown in Figure 1.

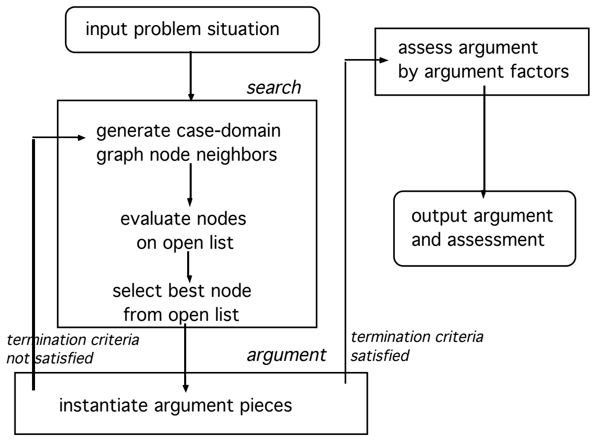
input problem situation
search
generate case-domain graph node neighbors
evaluate nodes on open list
select best node from open list
termination criteria not satisfied
argument
instantiate argument pieces
termination criteria satisfied
assess argument by argument factors
output argument and assessment

**Figure 1.** Control flow of the BankXX system.

In general, state-space search is defined by a triple: *initial state, set of operators on states, set of goal states.* In best-first search, an *evaluation function* is also used to guide the exploration of the state-space [Barr et al., 1981]. Figure 2 summarizes BankXX's application of the best-first search model, an overview of which is given here, and details furnished in subsequent sections.

The Search Space. In BankXX it consists of a semantic network whose nodes represent cases and legal theories from the application domain, the "good faith" issue for personal Chapter 13 bankruptcy plans, and labeled links represent their interconnections. We refer to this network as the *case-domain graph.* It is described in detail in Section 3.2.1.

The Start Node. In BankXX the default for the initial state is the user-supplied problem situation, which is represented using the same set of hierarchical frames used to represent a case as a collection of facts. Alternatively, the user can input the problem case but specify another node as the start node, for instance, a favorite or well-known case, like the *Estus* case, if one is known, in order to concentrate search initially in a particular region of the space. In a companion article, we address empirically the question of the impact of start node selection on program performance.

The Operators. The set of operators used in BankXX are called *neighbor methods*. These use links in the case-domain graph to generate the “successor” nodes to be opened in search. BankXX has 12 neighbor methods. In general, they are more complex than the simple following of outward arcs from a given node. Some follow in-space or cross-space pointers in a straightforward way. For instance, *case-theory-neighbors* generates all the cases that have applied a particular theory. Others, similar to macro-operators, follow a fixed sequence of links. For instance *theory-case-theory-neighbors* finds all the theories applied by any of the cases the use the theory used in the current node. These are described in detail in Section 3.2.1.

Goal Nodes. As we have already said, there are none in BankXX. (Although one could use them in the BankXX architecture, if one so desired.)

Evaluation Function. BankXX can be run with any of three evaluation functions, each of which captures knowledge about legal information and/or legal argument. These are described in detail in Section 3.5.

**Search States:** Set of nodes in a case-domain graph representing either a case at some level of abstraction or a legal theory (Section 3.2).

**Initial State:** (1) Problem situation or (2) user-specified node in the case-domain graph (this Section).

**Operators on States:** Set of functions called *neighbor methods* that trace a single link or a sequence of links in the case-domain graph (Section 3.4).

**Goal States:** None (this Section).

**Termination Criteria:** (1) Empty open list, or (2) user-specified time bounds exceeded, or (3) user-specified space bounds exceeded (this Section).

**Heuristic Evaluation:** One of three linear evaluation functions at different levels of abstraction (Section 3.5).

**Figure 2.** Summary of the search model used by BankXX, with section references in this paper.

The search performed by BankXX differs from the usual applications in three ways:

1. the richness of node expansions through its so-called *neighbor methods*,
2. the absence of well-defined goal states,
3. re-evaluation of the nodes on the OPEN list in each search cycle.

All of these differences were motivated by requirements of our problem domain, legal reasoning.[10]

We do not include goal states in our model because of the difficulties inherent in defining an "argument goal" in a way that is consistent with our informal understanding of how humans develop and evaluate legal arguments. It is hard to say in general that an argument does or does not meet some plausible persuasive or rhetorical goal, or even that one has completed the supporting research.

We re-evaluate the nodes on the OPEN list because we want their heuristic evaluations to reflect the current state of BankXX's problem solving. This is especially important with certain types of evaluation functions in BankXX—particularly, the argument-piece evaluation function—that dynamically reflect an up-to-date view of the state of problem-solving. Such evaluation functions, in effect, change during the course of BankXX's problem-solving. For example, when adequate amounts of legal information of a particular kind have been gathered by BankXX—supporting cases, contrary cases, etc.—the value of harvesting more of the same kind of information decreases and the weight of the terms that represent their importance in the argument-piece evaluation function decrease as well.[11] In essence, BankXX has gathered enough of that type of information and needs to consider other types and this is reflected in the heuristic evaluation. Regardless of these differences however, the basic paradigm of BankXX is heuristic search.

BankXX builds up the content of the argument pieces by performing heuristic search in a network of domain knowledge. BankXX always begins its processing by analyzing the problem situation for applicable domain factors and computing a claim lattice, which partially orders the cases that have some of the same factors at work as the current problem. The best and most on-point cases are identified. These provide potential new nodes to be explored and are always the first nodes to be placed on the open list.

BankXX continues by performing a variation on the standard cycle of iterative, best-first search. Neighbors of the current node are generated using BankXX's neighbor methods. Their worth, as well as those already on the OPEN list, are calculated

---

[10]Note that the third difference—re-evaluation of opened nodes—is the only real departure from the classic heuristic, best-first algortihm, for instance, as given in *The Handbook of AI*, Vol 1, Chapter 2, Section C3a, p. 61. [Barr et al., 1981]. In the classic algorithm for graphs, only those nodes that have been opened previously (and thus, are already on OPEN), have their values recomputed.

[11]This is implemented through mechanisms such as the zeroing-out of terms in the argument-piece evaluation function. See Section 3.5, below.

according to the evaluation function. The "best" node on the open list—one with the maximum value under the evaluation function—is identified and then examined by each of the argument pieces in turn in order to determine if it can contribute to that component of the argument. Information that can be is harvested by the argument pieces and is appended to their data structures. This cycle continues until the search exceeds a user-specified time or space bound (e.g., 30 nodes closed), or until the open list is empty.

At the conclusion of the search, the argument is assessed in terms of the argument dimensions[12] and BankXX outputs the argument in a template structured by the argument pieces. In this way the information needed to build up the various argument pieces and ultimately the overall argument is acquired incrementally during the search (Figure 1).

The current implementation contains 54 cases from the bankruptcy good faith domain. These directly spawn 108 nodes in the case-domain graph, with one representation for each case as a factual situation and one representation of each case in terms of the legal factors applicable to it. There are 70 nodes that capture intercase citations, 19 prototypical story scripts (10 of which have associated cases that instantiate the script), and 18 legal theories. There are 27 domain factors and 20 inter-theory links of 9 types. The system runs on the Macintosh family of computers, and is written in Macintosh Common Lisp v.2.0 using CLOS. We have used graphing facilities built at the University of Massachusetts that allow the user to examine the case network from a variety of indexing perspectives.

### 3.2. Domain Knowledge Representation

#### 3.2.1. The Case-domain graph

The case base in BankXX consists of a semantic network whose nodes represent cases and legal theories, and whose labeled links represent connections between the nodes. We refer to this network as the *case-domain graph,* which consists of *case-domain-graph nodes* —also called *case-graph nodes* (CGN)—together with labeled link edges. There are six types of case-domain-graph nodes: One type represents legal theories and five represent legal cases in various perspectives proven useful to human legal reasoners. The five ways legal cases are represented are:

(1) as factual situations,
(2) in terms of various legal factors,
(3) as bundles of citations,
(4) as stereotypical stories or scripts, and
(5) by the measure of their prototypicality.

[12]In the work reported here we do not utilize this assessment. Our intent was to use it as feedback for applying methods of machine learning to improve the system.

Cases of like type can be grouped into *spaces*: *Case Citation Space*, *Legal Theory Space*, etc. Each space captures a particular type of knowledge and its natural interconnections. The spaces impart a structure similar to the partition of a blackboard application's working memory into spaces. Each space contains case-domain-graph nodes that represent cases (or legal theories) according to a particular perspective that has proven useful to human legal reasoners (Figure 3).

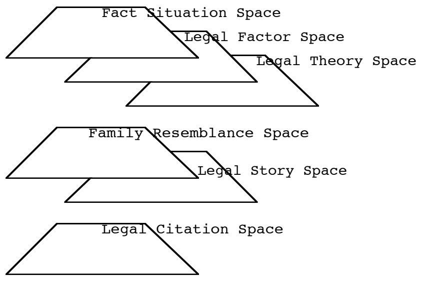

**Figure 3.** Spaces in the case-domain graph.

Nodes in the case graph are highly interconnected: *in-space* links connect objects within a space and *cross-space* links connect objects in different spaces. During search of the case graph, links are traversed by BankXX's neighbor methods, operators that expand nodes in the graph by following either in-space links, cross-space links or a sequence of links. Traversing a link is tantamount to using the link label as an index.

We now describe each space of case-domain-graph nodes, including its in-space links and some cross-space links. Additional cross-space links are described in Section 3.2.

**Fact Situation Space.** Fact situation space case-domain-graph nodes—or simply *case nodes*—encode legal cases as sets of facts. A case node, which is the representation in which a case is input, is the surface level of factual description and in many ways is the "generic" representation of a case. Each case node is represented as a tree of frames implemented as Common Lisp Object System instances. Examples of frames at this level describe the proposed plan and payments, the debt, the debtor's income, and generic information about the case. Cases at this level of representation are linked to each other through case citations. An example of a top-level frame for a case (the *Estus* case) as a fact situation node is given in Figure 4.

```
(make-instance 'STUDENT-LOAN-CASE
 :name 'ESTUS
 :case-link 'ESTUS
 :citation "695 F.2d 311 (8th Cir. 1982)"
 :year 1982
 :level :court-of-appeals
 :judge 'henley
 :summary  "Holder of VA student loan claim appeals confirmation of 15-month
 plan that gives zero payment to unsecured creditors.  Henley, J. held that good-faith
 requirement does not require <<substantial payment>> to unsecured creditors and lists
  meaningful factors.  Reversed and remanded."
 :procedural-status 'appeal-of-plan-confirmation
 :decision-for  'creditor                  ;appellant, reversed and remanded
 :citations  nil
 :factual-prototype 'student-loan     ;$2900 of $11000 unsecured debt is student loan
 :alternative-factual-prototype  nil
 :legal-prototype  nil
 :legal-theory '(ESTUS-THEORY)
 :chapter 13
 :plan-confirmed :no
 :estus-factors  'ESTUS-ESTUS-FACTORS
 :debt  'ESTUS-DEBT
 :plan-payments  'ESTUS-PLAN-PAYMENT
 :past-filings  nil
 :ch-7-filing-date  nil
 :plan-filing-date  "07-09-80"
 :unfair-manipulation  nil
 :attempts-to-pay  nil
 :profession  'federal-employee
 :dropout  nil
 :change-in-field  nil
 :loan-due-date nil
 :makarchuk-factors 'ESTUS-MAKARCHUK-FACTORS)
```
**Figure 4.** Example of top-level case frame for the Estus case.

**Legal Factor Space.** Legal cases can be represented in terms of their values on domain dependent factors or “dimensions” [Rissland, Valcarce & Ashley, 1984], [Ashley, 1990]. Factors are derived features recognized by domain experts as strongly influencing a case’s outcome. A factor allows cases to be compared and assessed as stronger or weaker with respect to the factor’s perspective. In Legal Factor Space, a case is represented by a vector composed of the magnitudes of the case on each dimension that applies to it; non-applicable factors are encoded as NIL. This vector of domain factor values represents a case as a point in an n-dimensional space. While there are no explicit links between nodes in the Legal Factor Space, the vector space structure of this space does allow us to define relations between them, for instance, according to how “close” they are. The specific sense of closeness depends on the choice of metric or distance function imposed on the space (e.g., standard Euclidean distance, cityblock distance). With respect to any given BankXX dimension, it is easy to derive relations between cases on that dimension, as was done in HYPO.

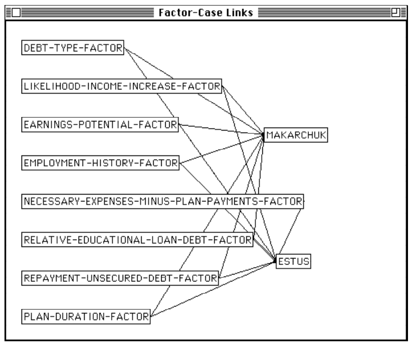
Factor-Case Links
DEBT-TYPE-FACTOR
LIKELIHOOD-INCOME-INCREASE-FACTOR
EARNINGS-POTENTIAL-FACTOR
EMPLOYMENT-HISTORY-FACTOR
NECESSARY-EXPENSES-MINUS-PLAN-PAYMENTS-FACTOR
RELATIVE-EDUCATIONAL-LOAN-DEBT-FACTOR
REPAYMENT-UNSECURED-DEBT-FACTOR
PLAN-DURATION-FACTOR
MAKARCHUK
ESTUS

**Figure 5.** A small subset of the indexing links between domain factors and cases.

Using a HYPO-style analysis [Ashley, 1990], BankXX creates cross-space links between factors and the cases to which they apply (Figure 5). Factor analysis of a problem case is one of the first steps in processing a new case. From the usual HYPO-style factor analysis, BankXX creates a regular claim lattice, from which it computes most on-point and best cases.

BankXX uses 24 factors in its domain. They are divided into three groups. The first comes directly from the *Estus* case; some of the eleven factors in the opinion actually spawn several of the factors used in BankXX. The second comes from the *Makarchuk* case. The third comes from a variety of sources.

The *Estus* factors are:

- percent-surplus-of-income-factor
- employment-history-factor
- earnings-potential-factor
- likelihood-income-increase-factor
- plan-duration-factor
- plan-accuracy-factor
- inaccuracies-to-mislead-factor
- preferential-creditor-treatment-factor
- secured-claims-modified-factor
- debt-type-factor
- nondischarge-7-factor
- special-circumstances-factor
- frequency-relief-sought-factor
- motivation-sincerity-factor
- trustee-burden-factor

The *Makarchuk* factors are:

- relative-total-payment-amount-factor
- relative-monthly-payment-amount-factor
- use-of-skills-gained-factor
- relative-educational-loan-debt-factor
- de-minimis-payments-factor

Other factors are:

- attempts-to-pay-factor
- repayment-unsecured-debt-factor
- necessary-expenses-minus-plan-payments-factor
- (formerly surplus-factor)
- unfair-manipulation-factor

**Figure 6.** The factors employed by BankXX.

**Legal Citation Space.** Citation case-domain-graph nodes encode the citations found in cases. Each citation case-domain-graph node represents one citing/cited pair: *Case-x citation-signal Case-y*. The *citation signal* specifies the sense in which a case is cited (*see* [Shepard's, 1994] [BlueBook, 1986][Ashley & Rissland, 1987]). Citation nodes are not linked to each other. Rather they provide (cross-space) links to the citing case and the cited case in Fact Situation Space (Figure 6) and thus indirectly link the citing and cited cases .

Citation space currently uses eight links: *accord, agrees-with, cites, discusses, eg, see, see-also, see-eg*. All but two (*agrees-with* and *discusses*) of their definitions correspond to their definitions as citation signals in *The Blue Book*:

(1) accord - Cited authority directly supports the proposition, but in a slightly different way than the authority(ies) first cited.
(2) cites - Cited authority (i) states the proposition, (ii) identifies the source of a quotation, or (iii) identifies an authority referred to in text.
(3) eg - Cited authority states the proposition; other authorities also state the proposition, but citation to them would not be helpful.
(4) see - Cited authority directly supports the proposition. See is used instead of "[no signal]" when the proposition is not stated by the cited authority but follows it.
(5) see-also - Cited authority constitutes additional source material that supports the proposition.
(6) see-eg - Cited authority provides other (helpful) support or illustration of the proposition. E.g. may also be used in combination with other signals
(7) agrees-with - The court opinion explicitly used the phrase agrees with (as in "We agree with the analysis in the Estus case…")
(8) discusses - same as intuitive notion of discusses: the court analyzed aspects of the precedent, usually devoting several paragraphs to a cases facts and/or previous analysis.

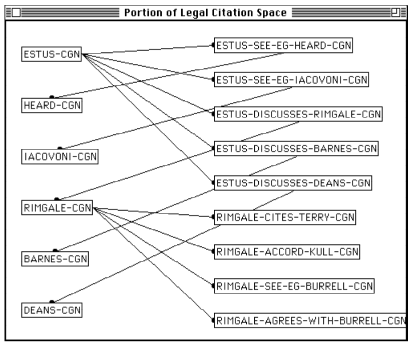
Portion of Legal Citation Space
ESTUS-CGN
HEARD-CGN
IACOVONI-CGN
RIMGALE-CGN
BARNES-CGN
DEANS-CGN
ESTUS-SEE-EG-HEARD-CGN
ESTUS-SEE-EG-IACOVONI-CGN
ESTUS-DISCUSSES-RIMGALE-CGN
ESTUS-DISCUSSES-BARNES-CGN
ESTUS-DISCUSSES-DEANS-CGN
RIMGALE-CITES-TERRY-CGN
RIMGALE-ACCORD-KULL-CGN
RIMGALE-SEE-EG-BURRELL-CGN
RIMGALE-AGREES-WITH-BURRELL-CGN

**Figure 7.** A small subgraph of the case graph, showing cases in Fact Situation Space (left side of figure) linked to citation nodes in Legal Citation Space (right side). "CGN" denotes a case-graph node.

**Legal Story Space.** In bankruptcy, as well as in other domains, cases often follow certain standard scripts or story lines.

The four bankruptcy story prototypes used most frequently by BankXX are:

(1) *the student loan story* — student incurs educational debts and soon after graduating files for bankruptcy protection from his educational loan creditors.

(2) *the dishonest debtor* — debtor commits fraud or some other offense, a judgment is entered against debtor, debtor files for bankruptcy.
(3) *the automobile debtor* — debtor borrows money to purchase automobile beyond his or her means, and declares bankruptcy to avoid repaying the portion of the loan above the value of the car.
(4) *the consumer debtor* — debtor purchases goods and services on credit, usually with credit cards, and files for bankruptcy to discharge the liabilities.

BankXX does not link story prototypes to each other. Exploiting such links would require an understanding of how stories can be related and, ideally, an automated means to recognize them (e.g., plot units [Lehnert, 1981]).

**Family Resemblance Space.** We have begun to incorporate some of the research of Rosch, who proposed a model of the internal structure of categories that is captured in the family resemblance hypothesis: "the most prototypical members of categories are those with most attributes in common with other members of that category and are those with least attributes in common with other categories" [Rosch and Mervis, 1975, p.576]. While Rosch proposed the family resemblance hypothesis as a cognitive structural model, we are experimenting with family resemblance as an indexing and processing model. BankXX can calculate the degree of family resemblance of a case to a given set of cases and select the cases within that family having the greatest family resemblance.

For instance, the system can calculate the family resemblance of all student loan cases, and find the most prototypical. For example, within the family of student loan cases *In re Ali*, 33 B.R. 890 (1983) is the most prototypical. *Ali*, involved a Kansas couple who proposed a repayment plan that would pay a loan secured by their car, but would pay nothing to unsecured creditors such as the University of Kansas, which had extended education loans to Mr. Ali. On the basis of an examination of a set of factors from *Flygare v. Boulden*, 709 F.2d 1344 (10th Cir. 1983), the court found that the Alis had proposed their plan in good faith.

**Legal Theory Space.** Legal theories are defined by a list of factors (see the discussion of Legal Factor Space) that are prerequisite conditions for a theory to apply to a case. Figure 8 shows the case-domain-graph node representing the *Estus* theory. Note that a theory node lists the case node (in Fact Situation Space) that promulgated the theory as well as those cases that actually applied it.[13] Thus the *Estus* case promulgated the ESTUS-THEORY and *Estus*, *Flygare* and *Makarchuk* actually applied it.

[13]Our criteria for listing a case as applying a theory were quite strict. For instance, we checked the opinion to see if the court considered the factors defining it. Just mentioning a theory was not sufficient reason to list it as applied.

```
(make-instance 'legal-theory
        :name 'ESTUS-THEORY
        :other-names '(FLYGARE-THEORY)
        :description "<omitted>"
        :factors '(surplus-factor
              percent-of-repayment-factor
              employment-history-factor
              earnings-potential-factor
              plan-duration-factor
              plan-accuracy-factor
              preferential-creditor-treatment-factor
              secured-claims-modified-factor
              debt-type-factor
              nondischarge-7-factor
              special-circumstances-factor
              frequency-relief-sought-factor
              motivation-sincerity-factor
              trustee-burden-factor)
        :factor-evaluation nil
        :domain-theories 'debt
        :view :majority
        :cases-promulgating '(estus)
        :cases-applying '(estus flygare makarchuk)
        :cases-rejecting nil
        :courts-adopting :8th-circuit                 )
```

**Figure 8.** Representation of the Estus theory.

Legal theory nodes are linked by pointers that describe the relationships between them (Figure 9), such as "overlaps with," "rejects," and "agrees with." Figure 9 shows some of the legal theories connected with the *Estus* case. Legal theories have been culled from opinions by hand. They have to have been mentioned explicitly in an opinion as a theory that has been applied or that will apply to the instant case in order for it to be represented as a theory within BankXX.

Currently, BankXX uses nine different linkages between theories:

(1) *overlaps-with,*
(2) *conflicts-with,*
(3) *rejects,*
(4) *derives,*
(5) *is-derived-from,*
(6) *agrees-with,*
(7) *refines,*
(8) *is-refined-by,*
(9) *is-equivalent-to*.

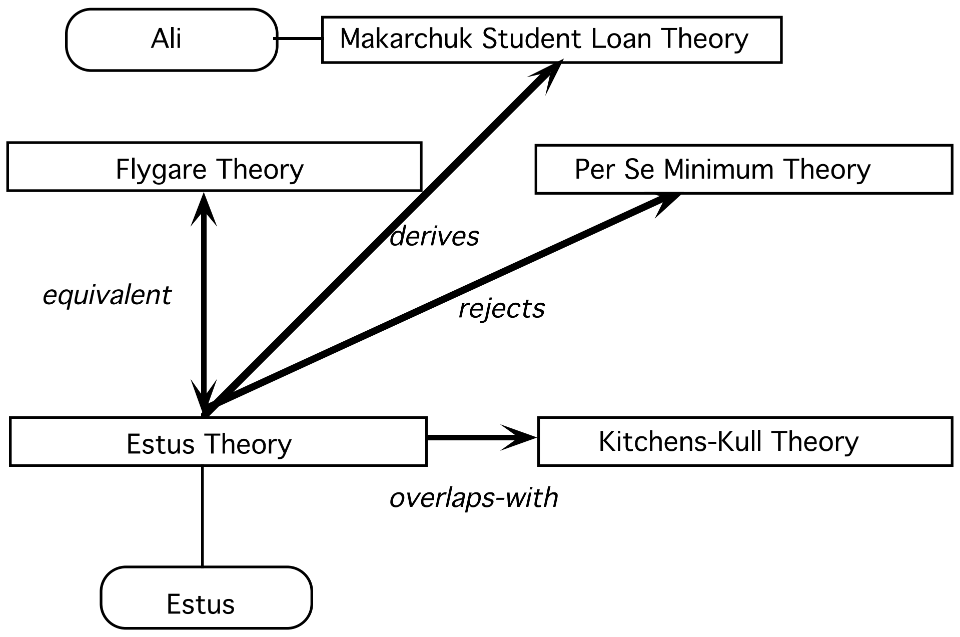

**Figure 9.** A small subgraph of the case-domain graph, showing inter-theory links and links from theories to cases.

In some preliminary research that is not incorporated into BankXX, we have begun to build a taxonomy of methods that manipulate legal theories. Theories may be manipulated by adding factors, limiting the factors considered, changing the way the factors are combined (as by a weighting scheme), or simply shifting the burden of persuasion of proving the theory. There is much research that can be directed at theory formation in the law.

### 3.2.2. Cross-Space Case Links

In addition to the spaces with their in-space links just described, a variety of bi-directional, cross-space links exist. For instance, links exist between factors and legal theories that use those factors, and between story prototypes and cases instantiating them. See Figure 10.

| | case | citation | theory | factual-prototype | domain-factor |
|---|---|---|---|---|---|
| case | x | x | x | x | x |
| citation | x | | | | |
| theory | x | | x | | x |
| factual-prototype | x | | | | |
| domain-factor | x | | x | | |

**Figure 10.** Table of links present between case-graph node classes in BankXX.

## 3.3. Argument Knowledge Representation

### 3.3.1. Argument Pieces: Some Building Blocks for Argument

We have chosen a simple representation of an "argument" for purposes of this implementation. In this application, an argument is a collection of *argument pieces*, that represent fragments of arguments or pieces of legal knowledge that an advocate would ideally like to have to support his position. The argument pieces represent building blocks of argument. We recognize that this idealization of argument does not reflect the logical and rhetorical connections between the various pieces of an argument, or the complexity of argument in general. BankXX's task is to gather information necessary to fill in these building blocks.

The 12 argument pieces currently used in BankXX are:

1. **Supporting Cases -** cases decided for the same side as the viewpoint assumed in the current problem situation. Synonymously called *pro, same-side* or *favorable* cases.
2. **Best Supporting Cases** - the best cases decided for the current viewpoint.
3. **Contrary Cases -** cases decided for the opposing side.
4. **Best Contrary Cases** - defined similarly to 2.
5. **Leading Cases** - the five most frequently cited cases in the BankXX corpus.
6. **Supporting Citations** - citations found in favorable cases that lead to other favorable cases.
7. **Factor Analysis** - the set of domain factors ("dimensions") that are applicable to the current problem situation.
8. **Overlapping Cases -** cases sharing a large proportion (75%) of domain factors.
9. **Applicable Legal Theories** - if each factor defining a theory is applicable to the problem situation, the theory is considered applicable.
10. **Nearly Applicable Legal Theories** - a theory is nearly applicable if a threshold percentage (50%) of defining factors apply.
11. **Factual Prototype Story category** - the category of story that the debtor's factual situation falls under.
12. **Family Resemblance Prototype** - favorable cases having the highest family resemblance rating, with respect to a given family, to the instant case according to the Rosch measure of family resemblance [Rosch & Mervis, 1975].

Each of these argument pieces is defined computationally in BankXX. The definitions of "most on-point," "best," and domain "factor" are based directly on those used in the previous systems, such as HYPO and CABARET, occasionally with some modification. For each argument piece, there is a "functional predicate" that determines if a node can supply that useful piece of an argument and a data structure containing an object slot to store entities that satisfy its predicate. For example, the *overlapping cases* argument piece has a predicate to determine if a case shares with the problem situation more than 75% of the factors found in the current problem. BankXX builds up their content incrementally (as its search proceeds) and the collection of all argument pieces is output to the user at the conclusion of BankXX's processing. There is no argument text generation facility within BankXX, however.

We describe each argument piece in a bit more detail.

**1.** ***Supporting-Cases.*** The cases that were decided for the same "side"—for the debtor or the creditor—as the viewpoint assumed in the current problem situation. Cases in which the plan was confirmed were considered decided for the debtor; if the plan was not confirmed, the decision was considered as for the creditor.

**2.** ***Best-Supporting-Cases.*** The "best" cases decided for the viewpoint assumed in the current problem. BankXX uses a variation on the definition of best case used in HYPO [Ashley, 1990]. One requirement of a best case in HYPO was that it be a most on-point case. In some situations, this means that a side may have no "best" cases. In order to assure that each side has at least one "best" case, BankXX includes as "best cases" those supporting cases that are most similar to the problem situation, whether or not they are also most on-point.[14] Also, best cases in HYPO depended on a sense of whether each dimension favored one side or the other, which is absent from the BankXX implementation.

**3.** ***Contrary-Cases.*** A case is contrary if it was decided opposite from the viewpoint of the problem situation.

**4.** ***Best-Contrary-Cases.*** Cases decided for the opposing viewpoint that are best cases in the sense described above.

**5.** ***Leading-Cited-Cases.*** Leading cited cases are the five cases cited most frequently in the full text opinions of the cases in the BankXX case base. These were identified by analyzing the full text of each of the opinions for the cases in the BankXX case base, extracting the citations, and counting the number of cites to each case.[15] The leading cited cases in order are:

1. *Rimgale,*
2. *Estus,*
3. *Goeb,*
4. *Deans,*
5. *Iacovoni.*

We note that there is significant overlap between the cases that one might intuitively identify as "leading" in this area and the most frequently cited cases according to this analysis.

**6.** ***Supporting-Citations.*** In BankXX these are defined in the following technical sense: they (i) are found in cases with the desired viewpoint, (ii) point to other cases with the same viewpoint, and (iii) use a "citation signal" indicating that the citing

[14]Compositionally these definitions of best case can be expressed as follows. In HYPO, best = same-side(mopc (claim-lattice)). In BankXX, best = mopc (same-side (claim-lattice)). The operations of selecting the same-side cases and the most on-point (maximal) cases do not necessarily commute. The only way that BankXX can fail to have a best case for a viewpoint is if there are no same-side cases for the viewpoint.

[15]Multiple citations within an opinion were not taken into account.

case agrees with the cited case (i.e., *accord, agrees-with, cites, see-eg*) (See Section 3.2.1). For example, the citation "*Estus* agrees with *Flygare*" would be considered supporting if three conditions hold: (1) the citing case (*Estus*) is decided for the current viewpoint; (2) the cited case (*Flygare*) is also decided for the current viewpoint; and (3) the citation signal ("agrees with") indicates that the citing case and cited case are compatible on this point. The usual direct sense of supporting citation as a "cite to a supporting case" is already covered by other argument pieces (i.e., *supporting cases, best supporting cases*), and thus the implementation uses this indirect definition in order to capture citations that were found even though the cited case has not been analyzed (i.e., "closed") by BankXX.

**7.** ***Factor-Analysis*****.** The set of domain factors that are applicable to the current problem situation. This argument piece is instantiated before commencing the search of the case-domain graph. The factor analysis is used to create a claim lattice and to determine if a legal theory is applicable when it is removed from the open list.

**8.** ***Overlapping-Cases*****.** Cases that share a large percentage of the factors with the problem cases that show substantial similarity to the problem situation. These cases may not be most on-point cases, but deserve to be captured by research, since they are potentially useful if the most on-point or best cases fail to pan out. The current implementation requires that a case share with the problem situation at least 75% of all the factors applicable to the problem situation.

**9.** ***Applicable-Legal-Theories.*** Each theory is defined in terms of a set of domain-level factors. If all of these prerequisite factors are applicable to the problem situation, then the theory is considered applicable. This is analogous to HYPO's definition of applicable dimension with respect to a current fact situation. See Figure 8 for an example of a legal theory.

**10.** ***Nearly-Applicable-Legal-Theories.*** A theory is nearly applicable to a case if a user-defined threshold percentage of factors defining the theory are applicable to the problem situation. The default threshold in BankXX is 50%.

**11.** ***Factual-Prototype-Story-Category.*** These are the story categories that the debtor's factual situation falls under. Each factual story prototype encountered during the search of the case-domain graph is added to this argument piece. The current implementation does not filter these nor does it attempt to determine which of them are in fact appropriate in the current problem case. Story categories were assigned by hand to the cases in the case base. The factual prototype story category is a first attempt to capture episodic knowledge in a case, that is, a general factual pattern that it entails.

**12.** ***Family-Resemblance-Prototype.*** The favorable cases that have the highest family resemblance rating to the given case, according to the Rosch measure of family resemblance [Rosch & Mervis, 1975], with respect to a given family. This argument piece is not fully utilized by the current implementation.

### 3.3.2. Argument Factors

Just as cases may be indexed and compared on the basis of domain factors [Rissland, Valcarce & Ashley, 1984], [Ashley, 1990], so may arguments be compared on the basis of "argument factors." *Argument factors* capture dimensions along which the quality of arguments may be compared and contrasted. They can aid the system in identifying the best arguments (e.g., by sorting arguments according to a partial order based on the factors that apply to an argument). The third type of evaluation function we have experimented with in BankXX is based on these factors; this function is also used to evaluate the final argument output by BankXX.

Eight *argument factors* are currently implemented in BankXX:
(1) *centrality-of-best-cases*,
(2) *centrality-of-theory*,
(3) *win-record-of-theory*,
(4) *win-record-of-theory-for-factual-prototypes*,
(5) *strength-of-best-case-analogies*,
(6) *factual-prototypicality-strength*,
(7) *strength-of-citations*, and
(8) *equally-on-point-cases*.

Briefly, the meaning of the argument factors follows.

**1.** ***Centrality-of-best-cases*** assesses centrality of the best cases retrieved for the argument. It determines how often those cases have been cited in other cases in the case base. BankXX computes a ratio: (a) the number of best supporting cases that are also leading cited cases, to (b) the number of best supporting cases. This factor is designed to capture the idea that better known best cases provide a stronger argument than obscure ones.

**2.** ***Centrality-of-theory*** determines how often the theory has been used, invoked, or compared to a theory used in a case. The idea is that an argument should rely on a well-known theory. Its value is the number of cases in the case base that have applied any of the theories in the applicable-theories argument piece.

**3.** ***Win-record-of-theory*** determines how often the applicable theories found supporting cases that have been associated with winning arguments for the current viewpoint. For each theory a ratio is computed: the ratio of (a) the number of cases in the case base that applied the theory and are also supporting cases for the current problem to (b) the number of cases in the case base that applied the theory. The value of the factor is the maximum of these ratios over all the theories applicable to the current problem case. The factor rewards theories that have been applied in cases decided for the current viewpoint. This factor rewards arguments citing applicable theories that have been associated with past winning arguments for the current viewpoint.

**4.** ***Win-record-of-theory-for-factual-prototypes*** determines the proportion of cases in which the theory has been successfully used in a case whose facts follow a recognized, stereotypical pattern, such as a student-loan case. For each theory a ratio is computed: the ratio of (a) the number of cases in the case base that (i) exhibit a factual prototype specified in the problem situation, (ii) apply a theory, and (iii) are decided for the current viewpoint to (b) the number of cases in the case that apply that factual prototype and that theory. The value of the factor is the maximum of these ratios computed for each applicable theory. This factor, which is a version of win-record-of-theory specifically aimed at factual prototypes, determines the extent to which a theory has been successfully relied on in cases that follow a particular stereotypical pattern.

**5.** ***Strength-of-best-case-analogies*** is implemented currently in terms of the number of legal factors that are in common to both the best cases cited in the argument and the current problem. Since it is possible for the cases in the best-cases argument piece to share only a few factors in common with the problem situation, a measure of the number of factors in common is desirable. Best cases that have more factors in common with the current case, all other things assumed equal, are often better cases to use in argument (for instance, because they allow a wider band of analogies to be made). This factor computes the maximum across all the best cases of the ratio of (a) the number of factors in common between a best case and the problem to (b) the total number of domain factors that have been defined.

**6.** ***Factual-prototypicality-strength*** computes a normalized family resemblance rating for the current problem, to determine how prototypical it is for cases of a particular story prototype. See the appendix for a description of the computation of factual prototypicality.

**7.** ***Strength-of-citations*** gives a measure of how often the cases mentioned in the *supporting citations* argument piece point to *leading cases* or *best cases*. The more citations from the supporting citations argument piece that cite leading cases or best cases, the better. Recall that a citation consists of a citing case, a citation signal, and a cited case. The value of this factor is the ratio of (a) the sum of (i) the number of cited or citing supporting cases that are leading cases and (ii) the number of cited or citing cases that are best cases to (b) the number of cases in the union of the best and leading cited cases.

**8.** ***Equally-on-point-cases*** measures the proportion of best cases for which there are no equally on-point cases for the opposing side that share the same subset of dimensions. It provides one indication of how many best cases may not be countered by the opposing cite with a contrary best case that is equally similar to the problem situation.

There are a large number of dimensions that we were able to identify along which arguments might be sorted, but did not (or could not) implement. Some suggestions might be *provenance of support* (cite good courts—and respected jurists—in the proper jurisdictions), *strength of argument type* (straightforward argument types may be considered more desirable than convoluted ones, such as the "turkey, chicken and fish" argument [Skalak & Rissland, 1992]), *abstractness* (appeal to principles or policies versus appeal to rules or cases), *consistency of support* (extent to which cited cases do not undermine propositions in an argument for which they have not been cited), *base of support* (reliance on many weak cases or a few good ones), and so forth.

### 3.4. Traversing the Case-domain graph - Neighbor Methods

We note that the case-domain graph as described thus far does not really constitute the search space searched by BankXX. While there are no more additional nodes in the search space than are present in the case-domain graph, there are additional edges between nodes. These edges are created by the neighbor methods, which determine to what nodes search may permissibly move to from a current position in the search space. These are described next.

The set of nodes to which a search algorithm may permissibly move in legal research is less constrained than in a classical search applications like game-playing since there are no universally agreed upon "rules" on how to generate "moves." In legal research, the number of "legal moves" is extremely large due to the immense variety and volume of legal knowledge and the ways in which it can be manipulated. In the law, there are numbers of sources of compiled knowledge to aid the attorney or legal assistant in the task of uncovering "moves" to be explored in researching an issue. For instance, Shepard's Citations gives inter-case and statute-case links for use in tracking down legal materials [Shepard's, 1992]. The West Publishing Company has developed a system of keys that index specific areas of legal practice. In addition, there are other, implicit links used in practice that are not reflected in standard materials: links that capture the fact that a case presents an instance of a typical, recurring fact situation, that is, a "story"; links between a case and the legal theory that is used to decide it, etc.

BankXX employs neighbor methods that use links from the case-domain graph to generate (successor) nodes as candidates for examination (i.e., place on the "open" list) in BankXX's search of the graph. Neighbor methods are of three types. They:
(1) return nodes that are one intervening case-domain graph link away, typically by following all outbound links of a specific type from the node;
(2) return nodes that are more than one case-domain graph link away, typically by moving to another space and then back again, or
(3) return nodes that are connected by dynamically-created links.
The neighbor methods contain the knowledge of how to move about in the case-domain graph. BankXX currently has 16 neighbor methods (though four of these were disabled for the experiments) to exploit the case-domain graph's high degree of interconnectedness.

Neighbor methods are "methods" in the object-oriented programming sense of a function that applies only to objects of a specific class. Depending on the class or type of the current node (case, domain factor analysis, legal theory), only those neighbor methods that apply to instances of that class will be applicable.

**Type 1.** These methods take a case-domain-graph node as input and output all the nodes linked to the input item via one in-space or cross-space link (of a specific type). For example, *legal-theory-link-neighbors* returns all the theory nodes that are linked to a given theory node by any of the nine in-space legal theory space links (e.g., *conflicts-with, rejects, refines* [see Section 3.2.1]). The neighbor method *legal-theory-case-neighbors* returns all the cases that apply a theory, that is, all the case nodes that are linked to the theory node (listed in the *legal-theory* slot of a case node) (See Figure 4.).

There are ten neighbor methods of type 1:

(1) *citation-neighbors*—returns all the citations (nodes in Case Citation Space encoding the citation signal, citing case, cited case, etc.) for a given case (via cross-space case-to-citation links).
(2) *cited-case-neighbors*—returns all the cases (nodes in Fact Situation Space) cited in a given case (via cross-space case-to-citation links).
(3) *specific-citation-neighbors*—returns the specific cited case (node in Fact Situation Space) in a given citation node (via cross-space citation-to-case links).
(4) *legal-theory-link-neighbors*—returns all the theories (nodes in Legal Theory Space) linked to a given theory (via in-space links in the Legal Theory Space).
(5) *legal-theory-case-neighbors*—returns all the cases (nodes in Fact Situation Space) that apply a given legal theory (via cross-space theory-to-case links).
(6) *legal-theory-neighbors*—returns all the theories (nodes in Legal Theory Space) applied in a given case (via case-to-theory links).
(7) *factual-prototype-case-neighbors*—returns all the cases that share a given factual prototype (via factual-prototype-to-case links).
(8) *factual-prototype-neighbors*—returns the factual prototype(s) for a given case (via cross-space case-to-prototype links).
(9) *domain-factor-analysis-neighbors*—returns the domain-factor-analysis node for a given case (via case-to-domain-factor links).
(10) *domain-factor-neighbors*—returns the domain-factor-analysis nodes whose applicable factors include the factors of the given domain-factor node.[16]

**Type 2.** Neighbor methods of type 2 are similar to macro-operators [Fikes, Hart & Nilsson, 1972] in that they collapse a series of link traversals into a single operator in order to perform a retrieval that has been recognized as useful by legal researchers. For example, the method *case-theory-theory-case-neighbors* starts with a case, follows the link—a case-to-theory cross-space link—to the first theory listed as applied in that case, then follows links—within the Legal Theory Space—from that theory to each theory that has been explicitly referred to in a favorable way (e.g., by an *agrees-with* link) by that theory, and then follows links—theory-to-case cross-spaces—back to the cases that have applied the favorably viewed legal theories. The *case-theory-*

[16]If the universe of factors were restricted to those found in a particular problem case (current fact situation), this method in effect would return the clusters of factors found in more on-point cases. This method simply examines set-subset relations between domain factor clusters.

*theory-case-neighbors* method returns cases that have applied similar theories, and provides a useful collection of cases to examine in the next stage of the search (Figure 11).

We refer to this process of moving from one space to another and then back again as a *boomerang* method. Type 2 methods are all boomerang methods. They permit indirect linking between nodes of a given type. The particular traversal path can be read directly from the method's name. For instance, *case-theory-theory-case-neighbors* moves from a case to a theory, to other theories, and then, back to other cases.

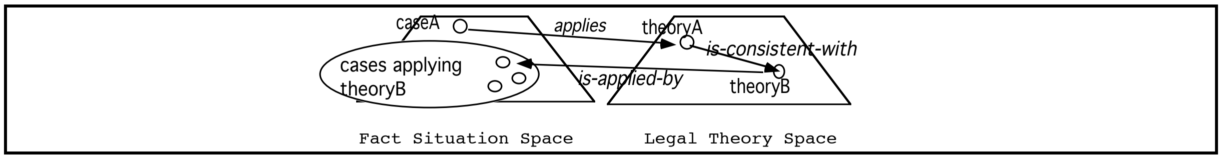

**Figure 11.** Tracing a series of links using the boomerang neighbor method *case-theory-theory-case*.

The four neighbor methods of type 2 are:

(1) *case-theory-theory-case-neighbors*—goes from a case to a theory applied in the case to other theories favorably viewed by the applied theory and then returns all the other cases that also applied to favorably viewed theories .

(2) *case-citation-citation-case-neighbors*—goes from a case to all its citations, then to other citation nodes, and back to case nodes. This method returns all cases citing a case cited in the input node.[17]

(3) *case-factual-prototype-case-neighbors*—goes from a case to its factual prototype and returns all the cases that also share that prototype.

(4) *factual-prototype-citation-neighbors*—goes from a factual prototype to all the cases that share the prototype, and then returns all the cases cited in any of those cases.

**Type 3.** There are two neighbor method of type 3. These methods create links dynamically. Type 3 neighbor methods take the problem case as a functional parameter, so that depending on the current problem different paths can be traced through the case-graph from a given node. The links followed from a given node in the case-domain graph depend on (1) the given node and (2) the context provided by the current problem situation. For example, the *family-resemblance-for-prototype-neighbors* method uses a measure of case prototypicality to link prototypical cases with other cases that fall within the same prototype family as the current problem.

The two neighbor methods of type (3) are:

(1) *family-resemblance-for-prototype-neighbors*—returns the case with the highest family resemblance score of those cases sharing its factual prototype.

(2) *family-resemblance-neighbors*—returns the five cases having the highest family resemblance score to the current problem situation.

[17]Another way to say this is: if Case-x cites Case-y, and Case-y cites Case-z, this function takes as input Case-x and returns all cases like Case-z.

## 3.5 Heuristic Evaluation Functions

BankXX uses three different types of evaluation functions. They differ in the level of abstraction used to evaluate case-domain-graph nodes. All are simple linear functions. They form a progression of increasingly more informed evaluation methods, whose considerations range from (1) only the type of information encoded in a node to (2) the contribution of a node to the standard argument pieces and (3) the incremental impact of a node on the overall state of the evolving argument, as assessed with the argument factors.

The three evaluation functions and their levels of abstraction are:

**1.** The *node-type evaluation function*—domain (node) level;
**3.** The *argument-piece evaluation function*—argument-piece level.
**2.** The *argument-factor evaluation function*—argument level

All three functions give greater weight to terms associated with legal theories. In our own experience as well as in reference works on research methods [Kunz, et al. 1992], there is evidence for the importance of legal theories in legal research. The five authors of this legal research methods text performed an experiment in which one of them posed a hypothetical research situation for the others to research. They recorded their activities in a protocol. One of the authors, a reference librarian, researched the case thoroughly. On the other hand, the remaining three authors "chose to stop when he or she determined that there were some legal theories to pursue on behalf of the client." [Kunz, et al. 1992, p. 454]. Thus biasing the BankXX evaluation functions toward legal theories appears to make good sense.

### 3.5.1. The Node-Type Evaluation Function

The form of the ***node-type evaluation function*** is $\sum w_i f_i(c)$. The $f_i$ check the *type* (e.g., legal case, legal theory) of the node $c$. The $w_i$ are non-negative, scalar weights. This function has the general form:

$$w_1 \; \textit{type-pred}_1(c) + w_2 \; \textit{type-pred}_2(c) + \ldots + w_n \; \textit{type-pred}_n(c)$$

Essentially the question asked by this evaluation function is: "How well will this node contribute information of a type known to be useful to argument?" It causes node-types to be examined in the order defined by the weights. Legal theories are given some preference but there is not much difference among the other types. In this implementation, we included only five terms; we did not use the prototypicality view of cases. The terms and term-weights used are given in Figure 12.

| | |
|---|---|
| 1.*Theory nodes* | 8 |
| 2. *Cases-as-facts nodes* | 6 |
| 3. *Citation-bundle nodes* | 5 |
| 4. *Domain-factor nodes* | 4 |
| 5. *Story Prototype nodes* | 3 |

**Figure 12.** Terms and weights used in the node-type evaluation function.

The node-type evaluation function is deliberately coarse-grained to provide a simple baseline. In order to provide slightly finer-grained distinctions for several types of case-domain-graph nodes, the implementation of BankXX applies additional tests to nodes of the citation-bundle, domain-factor and story-prototype types; these are described below. If the additional test is not satisfied by a node, the corresponding term in the node-type evaluation function is given 0 weight. If the additional test is satisfied, the weight given in Figure 12 is used. There are no additional tests for the theory nodes and the cases-as-facts nodes; the evaluation function tests only the type of those nodes, and the weights given in Figure 12 are always used for these terms.

<u>Citation-bundle nodes:</u> BankXX tests whether the citing case was decided for the point of view assumed in the problem case. The rationale for this additional test is that citations made in cases decided for the problem case's point of view are more likely to provide argument support than those cited in cases decided for the opposing side.

<u>Domain-factor nodes:</u> BankXX tests whether more than a threshold number of domain factors are applicable in the domain-factor node being evaluated. The rationale for this test is the heuristic that a "fat" domain-factor case analysis where a substantial number of factors apply is more likely to provide a useful piece of an argument than a "lean" one where only a few factors apply. The threshold is set at 10 applicable factors. (BankXX uses twenty-nine domain factors. See Appendix C.)

<u>Story-prototype nodes:</u> BankXX tests whether the story prototype of the problem case is the story prototype represented in the node being evaluated. The rationale is that story-prototype nodes representing other factual scenarios may not provide useful leads in researching a problem that has a different factual setting.

These tests yield somewhat finer distinctions in the evaluation of case-domain-graph nodes of the three types. To illustrate the story-prototype test, if the case-domain-graph node being evaluated is a *dishonest debtor* story-prototype node, but the problem case is a *student loan* case, then the story-prototype term would be assigned weight 0 because the story prototypes are different. If the story-prototype node being evaluated were *student loan* (as in the problem case) the weight for that term in the evaluation function would be 3.

### 3.5.2. The Argument-Piece Evaluation Function

The form of the ***argument-piece evaluation function*** is $\sum w_i f_i(c, a)$, where $c$ is the current node and $a$ is the current state of the argument. Each $f_i$ computes whether a particular *argument piece* is fillable by the current node and if that argument piece has *not* already been completely filled: if so, $f_i$ returns 1; else, 0. It is of the form:

$$w_1\ \textit{arg-piece-pred}_1(\mathrm{c,a}) + w_2\ \textit{arg-piece-pred}_2(\mathrm{c,a}) + \ldots + w_n\ \textit{arg-piece-pred}_n(\mathrm{c,a})$$

Essentially the question asked by this evaluation function is: “Can the particular domain knowledge contained in this node be used to fill one of the desired components of an argument that has not already been completely filled?” This intermediate-level evaluation function prevents BankXX from wasting computing resources by unnecessary bolstering of parts of the argument that are already well-established.

Note that all the predicates are computed with respect to the problem case, not with respect to the current node. So, for example, the applicable-theories argument-piece-predicate is only true (and returns 1) if the current node is a legal theory and it applies to the problem situation originally posed to the system.

In our experiments this evaluation function had only ten terms since we did not use the *family resemblance prototype* and *domain factors* argument pieces. Since the domain factor analysis is constant throughout a given problem case (it is computed but once, initially, at the beginning of the search), it simply contributes a constant value to the evaluation function; thus, we left it off. The actual terms, weights, piece limits (given in brackets) used are given in Figure 13.

1. *supporting cases*: weight=2 [limit=3]
2. *best supporting cases*: 7 [5]
3. *contrary cases:* 1 [3]
4. *best contrary cases:* 5 [3]
5. *leading cases*: 6 [5]
6. *supporting citations*: 1 [5]
7. *overlapping cases*: 1 [5]
8. *applicable legal theories*: 8 [6]
9. *nearly applicable legal theories*: 6 [3]
10. *factual prototype stories*: 6 [1]

**Figure 13.** Terms, weights, and fill limits (in brackets) for the argument-piece evaluation function.

As one can see, the argument-piece evaluation function is biased towards legal theories, best supporting cases, leading cases, and stories. Note that a case that fits into more than one category can have quite a high score due to the additive nature of the function. For example, a *leading-case* like *Rimgale* that in a given problem situation is also a *supporting-case*, would have an evaluation score of 8 (= 2 + 6). If it were also a best case, its score would be 15 (=2 + 7 + 6).

Note, that the value an item like a case receives depends on the state of BankXX’s problem-solving. For instance, suppose *Rimgale* is not a best case and that at some point in BankXX’s problem-solving, the *supporting-cases* argument piece has reached its fill limit of 3 cases, and the *leading-cases* argument piece has not, then the evaluation score for *Rimgale* would be only 6.

It is important to note that when BankXX is run with the argument-piece evaluation function, the fill limits on the various argument piece terms are also used to limit which items are actually harvested. It is possible for a case to be ignored by certain categories that it ostensibly fits because these categories are already filled with harvested information. For instance, in the *Rimgale* example of the previous paragraph, *Rimgale* would only be harvested—and listed in the output—under *leading-cases* even though it is also a *supporting-case*. Note that this filling up of a category is reflected in the evaluation score awarded a node: a filled-up argument piece has a zeroed out term in the evaluation function. Thus, one can say that information is harvested for only those argument pieces for which there is a compelling rationale to do so, that is, a non-zero contribution to the evaluation score. In the *Rimgale* example here, only the *leading-cases* term will contribute a non-zero value because the *supporting-cases* term will be zeroed out.

In summary, when BankXX is run with the argument-piece evaluation function, the fill limits on argument pieces affect BankXX in two ways:

(1) by zeroing out terms in the evaluation function for argument pieces filled to their limits, and
(2) by causing closed nodes not to be harvested by filled argument pieces.

### 3.5.3 The Argument-Factor Evaluation Function

The form of the ***argument-factor evaluation function*** is $\sum w_i f_i(c, a, a^*)$, where $a^*$ is the argument that would result from incorporating the knowledge in node $c$ into the current argument $a$. The $f_i$ compare the values along each of the *argument dimensions* applied to the current argument $a$ with those of the argument $a^*$. It is of the general form:

$$w_1\, \textit{arg-dim-fcn}_1(c,a,a^*) + w_2\, \textit{arg-dim-fcn}_2(c,a,a^*) + \ldots + w_n\, \textit{arg-dim-fcn}_n(c,a,a^*)$$

Essentially the question asked by this evaluation function is: "Can the domain knowledge contained in this node improve the quality of the argument?" Eight argument factors are used in this evaluation function. The actual terms and weights used in this evaluation function are given in Figure 14.

| | |
|---|---|
| 1. *centrality-of-theory* | 8 |
| 2. *win-record-of-theory* | 8 |
| 3. *win-record-of-theory-for-factual-prototype* | 8 |
| 4. *strength-of-best-case-analogies* | 5 |
| 5. *centrality-of-best-cases* | 5 |
| 6. *equally-on-point-cases* | 4 |
| 7. *strength-of-citations* | 4 |
| 8. *strength-of-factual-prototype* | 3 |

**Figure 14.** Terms and weights for the argument-factor level evaluation function.

## Part III: Extended Example

## 4. Example

To illustrate the workings of BankXX, we step through a run of BankXX on an actual legal case: *In re Estus,* 695 F.2d 311 (8th Cir. 1982). As we described in an overview of the domain, *Estus* is a leading case in this area. We give quite a bit of detail in describing the initial stages of the search in order to orient the reader to the system, and then describe the processing more generally at later stages.

### 4.1. The Estus Case

The facts of the case are these. The debtors, the Estuses, filed a Chapter 13 petition in 1980, listing almost $11,000 in debts to some 30 unsecured creditors, including almost $3000 in student loans from the appellant, the U.S. government, which was a holder of an unsecured claim arising from a Veterans Administration educational loan. The Estuses also owed secured debts to a furniture and a piano store. They were also five months in arrears on the mortgage on income-producing rental property they owned. The Estuses had a monthly income of $745, and monthly expenses of $492, leaving a surplus of $253. They proposed to pay $250 each month, all of which was to be applied toward the two secured debts and the mortgage payments. No payments were to be made under the plan to any of the unsecured creditors. The term of the proposed plan was only 15 months.

Procedurally, this case was appealed by the U.S. government from a decision of the U.S. District Court for the Eastern District of Arkansas, upholding the Bankruptcy court's confirmation of the debtors Ronald and Doris Estus's Chapter 13 plan.

### 4.2. Running BankXX on Estus

We treat *Estus* as a new fact situation, which we call the **Estus-problem** case. In order to treat *Estus* in a *de novo* manner, the real *Estus* case was excised from the case-domain graph. We did retain the node for the so-called *Estus* theory; however, there is no connection in the case-domain graph between the Estus-problem case and the *Estus* theory: it is as though the *Estus theory* is a legal theory that arose from some other case.

We configured BankXX with the argument-piece evaluation function and the following system parameters: close at most 30 nodes; use no more than 1,000 billable seconds;[18] start at a most on-point case for Estus-problem case; take the point of view of the creditor, the U.S. government. We note that fill limits (e.g., 3 cases each for *supporting-cases* and *contrary-cases*) will play a role in the extended example.

[18]"Billable seconds" is the time BankXX takes from the moment it is invoked until it returns its output. It includes any time spent on garbage collection and output to the screen during intermediate processing.

## 4.3. Extracts from the Processing of Estus

Before search of the case domain graph begins, the problem case is analyzed by the HYPO-based part of BankXX, which determines which of the factors applies to it. A claim lattice [Ashley, 1990] is built based on the analysis of the current fact situation and upon a factor analysis of each of the cases in the case-domain graph. The factor analysis is stored in the *factor-analysis* argument piece, whose sole function is to hold the results of that analysis.

The factors that are applicable in the Estus-problem case are:

- preferential-creditor-treatment-factor
- nondischarge-7-factor
- percent-surplus-of-income-factor
- plan-duration-factor
- repayment-unsecured-debt-factor
- relative-educational-loan-debt-factor
- necessary-expenses-minus-plan-payments-factor
- employment-history-factor
- earnings-potential-factor
- likelihood-income-increase-factor
- debt-type-factor

Creating the claim lattice allows the system to extract the most on-point cases from the claim lattice for the Estus-problem case: *Akin*, *Gunn*, and *Gibson*. See Figure 16.

One case is chosen at random from among the most on-point cases. In this example, *Akin* is chosen. It is added to the open list. See Figure 15. This completes the initialization of BankXX on the *Estus* problem.

```
cases:
[AKIN]
```

**Figure 15.** The open list at the start of the first cycle for the Estus-problem case.

With one case to start with on the open list, BankXX enters the basic iterative portion of its execution cycle.

(1) evaluate each node on the open list and select one with the maximum value. Remove this maximal node from the open list and place it on the closed list;
(2) apply the predicate of each of the argument pieces to the maximal node, and add the node to each argument piece whose predicate is satisfied;
(3) generate the neighbors of the maximal node using the neighbor methods, and place them on the open list of potential nodes to visit.

This three-stroke cycle is then repeated, selecting a new maximal node each time.

Since [AKIN] is the only node on the open list to start with, the first step is trivial in this first cycle. [AKIN] is selected as the maximal node and moved to the closed list.

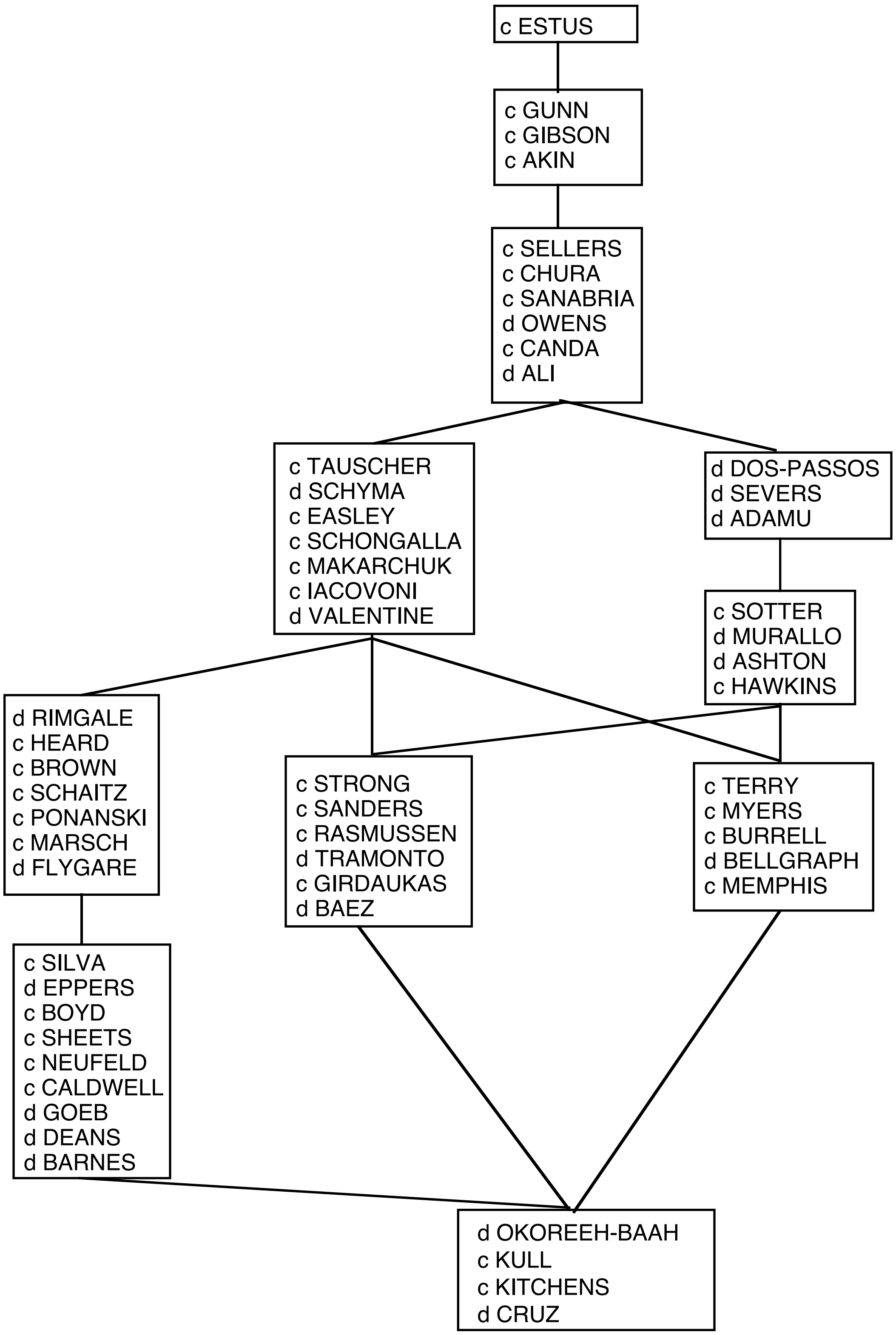

**Figure 16.** The claim lattice for the Estus-problem case. A case won by the creditor is indicated with a ***c***; a case won by the debtor with a ***d***.

The second step is for each one of the twelve argument pieces to examine [AKIN] in turn to determine if information can be harvested from [AKIN] that can be used to fill that piece. *Supporting-cases*, which is a repository for cases that were decided for the same vantage point (the creditor) as the side taken by BankXX in the current case, can use [AKIN] simply because it was decided for the creditor. *Best-supporting-cases* uses [AKIN] because it is a most on-point case that has been decided for the creditor, and therefore is a best supporting case. [AKIN] is thus harvested by both *supporting-cases* and *best-supporting-cases.* Both can glean information from [AKIN]; the other argument pieces cannot. For instance, because *Akin* is not one of the top five leading cases, the *leading-cited-case* argument piece passes over [AKIN]. Because [AKIN] does not share 75% or more of the Estus-problem case's domain factors, the *overlapping-cases* argument piece does not harvest [AKIN]. As a result of the first search cycle, the argument appears as in Figure 17.

```
SUPPORTING-CASES: ([AKIN])
BEST-SUPPORTING-CASES: ([AKIN])
CONTRARY-CASES: NIL
BEST-CONTRARY-CASES: NIL
LEADING-CASES: NIL
SUPPORTING-CITATIONS: NIL
FACTOR-ANALYSIS: ([ESTUS-PROBLEM-FACTOR-ANALYSIS])
OVERLAPPING-CASES: NIL
APPLICABLE-LEGAL-THEORIES: NIL
NEARLY-APPLICABLE-LEGAL-THEORIES: NIL
FACTUAL-PROTOTYPE-STORY-CATEGORY: NIL
FAMILY-RESEMBLANCE-PROTOTYPE: NIL
```

**Figure 17.** Argument after one search iteration on the Estus-problem case.

In the third step BankXX uses the maximal node to locate new nodes to open. To execute this step, BankXX applies all of its neighbor methods to the maximal node.[19] Six neighbor methods apply to [AKIN], five of which yield new nodes. The remaining neighbor methods do not apply to [AKIN]. The details are as follows:

The *citation-neighbors* method applies, yielding two citations, [AKIN-CITES-ESTUS] and [AKIN-SEE-ESTUS]. However, since both citations cite *Estus*, they cannot be used.[20]

The *specific-citation-neighbors* method does not apply to [AKIN].

The *cited-case-neighbors* method follows the case-to-citations links from the AKIN case to the citation nodes [AKIN-CITES-ESTUS] and [AKIN-SEE-ESTUS]

[19]Five neighbor methods were disabled to simplify the exposition of this example: the two family resemblance methods, *family-resemblance-for-prototype-neighbors* and *family-resemblance-neighbors*, and three of the boomerang methods, *case-citation-citation-case-neighbors*, *case-factual-prototype-case-neighbors*, and *factual-prototype-citation-neighbors,* all of which can return a large number of neighbors.

[20]While most linkages can be removed before processing a case in a *de novo* manner, realistically some information can be 'removed' only when it is run across by BankXX.

and returns the cases from the citations. Both the citations yield the *Estus* case. But since *Estus* is the problem case and therefore is not an admissible node in the search, [ESTUS] is not added to the open list.

The *legal-theory-case-neighbors* method does not apply to [AKIN].

The *legal-theory-link-neighbors* method does not apply either.

The *legal-theory-neighbors* method returns the [ESTUS-THEORY] because it is represented as having been applied by the court in [AKIN].

The *factual-prototype-case-neighbors* method does not apply to [AKIN].

The *factual-prototype-neighbors* method yields the [STUDENT-LOAN] factual prototype because *Akin* was tagged as a student loan case.

The *domain-factor-neighbors* method does not apply to [AKIN].

The *domain-factor-analysis-neighbors* method returns the factor analysis of the *Akin* case, [AKIN-FACTOR-ANALYSIS], which it has computed.

The *case-theory-theory-case-neighbors* "boomerang" method looks at the cases that apply any of the theories that are linked to the theories applicable in the *Akin* case, which is the [ESTUS-THEORY]. The cases applying the *Estus* theory are [MAKARCHUK], [CRUZ], [BAEZ], [ASHTON], [BURRELL], [RASMUSSEN], [CHURA], and [ALI].

The *factual-prototype-citation* method does not apply to [AKIN].

The neighbor methods have discovered 11 nodes. All of these are placed on the open list. This concludes the first cycle.

At the start of the second cycle, there are now many nodes on the open list. Applying the argument-piece evaluation function to each of the nodes on the open list yields the open list with accompanying evaluations shown in Figure 18.[21]

theories:
[ESTUS-THEORY] 6

factual prototypes:
[STUDENT-LOAN] 6

factor analyses:
[AKIN-FACTOR-ANALYSIS] 1

cases:
[ALI] 6, [MAKARCHUK] 2, [CRUZ] 1, [BAEZ] 1, [ASHTON] 1, [BURRELL] 2, [RASMUSSEN] 2, [CHURA] 2

**Figure 18**. The open list at the start of the second cycle from the Estus-problem case.

BankXX picks the node with the maximum evaluation function value. The tie-break policy for choosing among nodes with maximal evaluations is to pick a random theory if a theory is present, and, if a theory is not present, to pick the maximal item

[21]Note applying the argument-piece evaluation function is not as onerous as it might appear; most of the terms can easily be computed by reference to the node or the claim lattice for the problem case.

that was most recently added to the open list. The [ESTUS-THEORY] is chosen. BankXX moves to it—making it the current node—from the [AKIN] node, which was closed in the first cycle. [ESTUS-THEORY] is placed on the closed list.

Having chosen [ESTUS-THEORY] as the new current node, the second step in this second cycle is for all the argument pieces to examine it. In fact, only the *applicable-legal-theories* and *nearly-applicable-legal-theories* argument pieces have to consider [ESTUS-THEORY] in any detail. Since the Estus-problem case does not present enough information to make a determination on all the factors underlying the *Estus* theory, [ESTUS-THEORY] cannot be used by the *applicable-legal-theories* argument piece. However since more than 50% of its prerequisite factors apply to the problem situation, it is used by the *nearly-applicable-legal-theories* argument piece. Figure 19 shows the emerging argument for the Estus-problem case after [ESTUS-THEORY] has been harvested.

SUPPORTING-CASES: ([AKIN])  
BEST-SUPPORTING-CASES: ([AKIN])  
CONTRARY-CASES: NIL  
BEST-CONTRARY-CASES: NIL  
LEADING-CASES: NIL  
SUPPORTING-CITATIONS: NIL  
FACTOR-ANALYSIS: ([ESTUS-PROBLEM-FACTOR-ANALYSIS])  
OVERLAPPING-CASES: NIL  
APPLICABLE-LEGAL-THEORIES: NIL  
NEARLY-APPLICABLE-LEGAL-THEORIES: (**[ESTUS-THEORY]**)  
FACTUAL-PROTOTYPE-STORY-CATEGORY: NIL  
FAMILY-RESEMBLANCE-PROTOTYPE: NIL

**Figure 19**. Estus-problem case argument after two search cycles Newly added nodes are in bold.

Having completed the harvesting step—in which the [ESTUS-THEORY] has been added to the *nearly-applicable-legal-theories* argument piece—the third step in this second cycle is to use [ESTUS-THEORY] to generate more nodes for the open list. The following neighbor methods are applicable to [ESTUS-THEORY]:

(1) The *legal-theory-link-neighbors* method yields [MAKARCHUK-PRINCIPAL-PURPOSE-STUDENT-LOAN-DISCHARGE-LEGAL-THEORY], [KITCHENS-KULL-THEORY], [PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY] and [FLYGARE-THEORY]. (See Figure 9.)

(2) The *legal-theory-case-neighbors* method returns a list of cases that have applied the [ESTUS-THEORY] in the past, including [CHURA], [FLYGARE], [SANDERS], [SCHYMA], [SELLERS], and [VALENTINE].

Thus when the various neighbor methods are applied to [ESTUS-THEORY], a set of other theories is added to the open list by virtue of the links between [ESTUS-THEORY] and these other theories. These are exploited by the *legal-theory-link-neighbors* method. Several new cases are also added. The open list just after additions is given in Figure 20. There are now 19 nodes on the open list.

theories:
**[FLYGARE-THEORY], [PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY], [KITCHENS-KULL-THEORY], [MAKARCHUK-PRINCIPAL-PURPOSE-STUDENT-LOAN-DISCHARGE-THEORY]**

factual prototypes:
[STUDENT-LOAN]

factor analyses:
[AKIN-FACTOR-ANALYSIS]

cases:
[ALI], [MAKARCHUK], [CRUZ], [BAEZ], [ASHTON], [BURRELL], [RASMUSSEN], [CHURA], **[VALENTINE], [SELLERS], [SCHYMA], [SANDERS], [FLYGARE],**

**Figure 20**. The open list at the end of the second cycle on the Estus-problem case. New nodes are shown in bold face.

In its third, fourth, and fifth cycles, the system closes in turn [KITCHENS-KULL-THEORY] and the [PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY], both of which are applicable theories (and have maximal scores of 8), and the [FLYGARE-THEORY], which is nearly applicable (and chosen before the [ALI] node, which also has a score of 6, because it is a theory). The theory from *Makarchuk* case is not closed because it fails to meet the threshold needed to be categorized as a *nearly-applicable-legal-theory* and thus has a low evaluation score (0). The bias of the argument-piece evaluation function for legal theories is evident here. After five nodes have been closed, the argument now is given in Figure 21. At this point, BankXX has discovered and harvested information from a region of legal theories that provides theories useful to an argument for the Estus-problem case.

SUPPORTING-CASES: ([AKIN])
BEST-SUPPORTING-CASES: ([AKIN])
CONTRARY-CASES: NIL
BEST-CONTRARY-CASES: NIL
LEADING-CASES: NIL
SUPPORTING-CITATIONS: NIL
FACTOR-ANALYSIS: ([ESTUS-FACTOR-ANALYSIS])
OVERLAPPING-CASES: NIL
APPLICABLE-LEGAL-THEORIES:
  (**[PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY]**
  **[KITCHENS-KULL-THEORY]**)
NEARLY-APPLICABLE-LEGAL-THEORIES: ([ESTUS-THEORY], **[FLYGARE-THEORY]**)
FACTUAL-PROTOTYPE-STORY-CATEGORY: NIL
FAMILY-RESEMBLANCE-PROTOTYPE: NIL

**Figure 21.** Intermediate argument for the Estus-problem case after five nodes have been harvested. Nodes that have been added since the end of the second cycle are shown in bold face.

The next node that is closed is the [ALI] case, which was placed on the open list by the *case-theory-theory-case* neighbor method back in cycle 1. This node turns out to be important, for *Ali* is a useful case in many respects. Without going through all the details of the processing, *Ali* is harvested by the *best-contrary-cases* argument piece since it is one of the maximally on-point cases of those decided for the debtor (see Figure 15) and thus is a *best* case according to BankXX's definition of *best* (see Section 3.3.1). It is also harvested by the ordinary *contrary-cases* argument piece. *Ali*, in addition to providing another citation for the [FLYGARE-THEORY], cites [BARNES], [KITCHENS], [GOEB], [RIMGALE] and [DEANS]. These cases are added to the open list by the *cited-case-neighbors* method.

After [ALI], the leading [RIMGALE] case is closed next. It is harvested by both *leading-cases* and *contrary-cases*. The [RIMGALE] case leads to a legal theory which applies the definition of "good faith" that appears in a precursor bankruptcy statute, the [OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION-LEGAL-THEORY]; it is closed and harvested next. This legal theory was discussed in the *Rimgale* case and is unearthed by the *legal-theory-neighbors* method, which provides pointers from a case to each of the legal theories discussed in the case. After incorporating [RIMGALE] into the *leading-cases* and *contrary-cases* argument pieces, the argument now appears as in Figure 22.

SUPPORTING-CASES: ([AKIN])  
BEST-SUPPORTING-CASES: ([AKIN])  
CONTRARY-CASES: (**[ALI], [RIMGALE]**)  
BEST-CONTRARY-CASES: (**[ALI]**)  
LEADING-CASES: (**[RIMGALE]**)  
SUPPORTING-CITATIONS: NIL  
FACTOR-ANALYSIS: ([ESTUS-FACTOR-ANALYSIS])  
OVERLAPPING-CASES: NIL  
APPLICABLE-LEGAL-THEORIES:  
    ([PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY]  
    [KITCHENS-KULL-THEORY],  
    **[OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION-LEGAL-THEORY]**)  
NEARLY-APPLICABLE-LEGAL-THEORIES: ([ESTUS-THEORY], [FLYGARE-THEORY])  
FACTUAL-PROTOTYPE-STORY-CATEGORY: NIL  
FAMILY-RESEMBLANCE-PROTOTYPE: NIL

**Figure 22.** Intermediate argument for the Estus-problem case after eight nodes have been harvested. Nodes that have been added since the end of the fifth cycle are shown in bold face.

At this stage the argument is beginning to be fleshed out quite well, with many of the argument pieces containing information. The next item to be harvested is the [GOEB] case, which is both a leading-cited case and a contrary case; it is harvested by both the *contrary-cases* and *leading-cases* argument pieces. This is the third contrary case that BankXX has harvested and it thus fills up the *contrary-case* argument piece, which has a fill limit of 3.

One piece that is lacking is a characterization of the case as a factual prototype. In the next cycle, the [STUDENT-LOAN] prototype, which was placed on the open list in cycle 1, becomes the maximal node, is closed, and is harvested by the *factual-prototype-story* argument piece. The neighbor method *factual-prototype-case-neighbors* yields the [GUNN] and [GIBSON] cases, which are placed on the open list.

Processing now continues in this vein with BankXX further filling out the argument pieces. Note that the [ALI] case has lead to a number of useful nodes: [RIMGALE], which was harvested as both a contrary and leading case, [GOEB], which was also harvested as both a contrary and leading case, [DEANS], another contrary and leading case, and [IACOVONI], which is both a supporting and a leading case. The notion that some cases (such as *Ali* here) are very rich in citations that turn out to be useful is a familiar experience in doing legal research.

After harvesting [GUNN] and [GIBSON], two more *supporting-cases* that are also *best-supporting-cases*, BankXX closes the *Deans* case. In contrast to [RIMGALE] and [GOEB] however, [DEANS] can only be harvested by the *leading-case* argument piece even though it is also a contrary case. This occurs because [ALI], [RIMGALE] and [GOEB] were previously harvested and the *contrary-cases* argument piece is already full. The analogous situation obtains with [IACOVONI], which is closed in the next cycle, since it cannot be harvested by the *supporting-cases* argument piece because its fill limit of 3 cases has already been reached by [AKIN], [GUNN] and [GIBSON].

After examining a number of leading cases, BankXX considers the factor analyses of some of the cases that had been previously opened. For the next 5 cycles, it closes domain-factor-analysis nodes. (See items 16 through 20 in Figure 24.) Factor analyses for cases that share a large percentage of the factors with the problem situation, but whose cases are not most on-point cases, are collected in the *overlapping-cases* argument piece.

In cycle 21, the system turns to a set of cases that are on the open list. See Figure 24. None of the cases closed as the 21st through 29th nodes ([MAKARCHUK] through [SANDERS]) are harvested for the emerging argument since the *supporting-cases* and *contrary-cases* argument pieces are already full. All have evaluation function scores of 0 due to the fact that none of these cases is remarkable and the ordinary *supporting-cases* and *contrary-cases* argument pieces are full. For instance, none of these cases is a best case; none is particularly high up in the claim lattice (Figure 15), except [CHURA] or possibly [MAKARCHUK]. Indeed, none of these cases were actually mentioned in the actual *Estus* opinion, so nothing is actually missed by BankXX by not harvesting them (see discussion below).

The last node closed in this example is a theory node that had not been examined before, the [ABUSE-OF-CHAPTER-13-LEGAL-THEORY], which turns out to be nearly applicable to the current problem and is added to the argument. It is harvested by the *nearly-applicable-legal-theories* argument piece. This is an important addition to BankXX's store of information for the Estus-problem case. It is one of the five theories explicitly applied in the actual *Estus* opinion.

## 4.4. Results of the Processing

The state of the argument at the end of the processing is given in Figure 23.

```
SUPPORTING-CASES: [AKIN], [GUNN], [GIBSON]
BEST-SUPPORTING-CASES: [AKIN], [GUNN], [GIBSON]
CONTRARY-CASES: [ALI], [RIMGALE], [GOEB]
BEST-CONTRARY-CASES: [ALI], [OWENS]
LEADING-CASES: [RIMGALE], [GOEB], [DEANS], [IACOVONI],
SUPPORTING-CITATIONS: NIL
FACTOR-ANALYSIS: [ESTUS-PROBLEM-CASE-FACTOR-ANALYSIS]
OVERLAPPING-CASES: [AKIN-FACTOR-ANALYSIS], [ALI-FACTOR-ANALYSIS],
            [GUNN-FACTOR-ANALYSIS], [GIBSON-FACTOR-ANALYSIS],
            [OWENS-FACTOR-ANALYSIS]
APPLICABLE-LEGAL-THEORIES: [KITCHENS-KULL-THEORY],
            [PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY],
            [OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION-LEGAL-THEORY]
NEARLY-APPLICABLE-LEGAL-THEORIES: [ESTUS-THEORY]
            [FLYGARE-THEORY],
            [ABUSE-OF-CHAPTER-13-LEGAL-THEORY]
FACTUAL-PROTOTYPE-STORY-CATEGORY: [STUDENT-LOAN-FACTUAL-PROTOTYPE]
FAMILY-RESEMBLANCE-PROTOTYPE: NIL
```

**Figure 23**. Argument pieces at conclusion of processing of the Estus-problem case.

BankXX has closed 30 nodes (the pre-set limit on this run), but it has opened a much larger number: 121 in this run. Of the 30 nodes closed, only 21 have actually been harvested by the argument pieces due to fill limits. While 121 is almost half of the nodes in the case graph, it is a testament to the perspicuity of the evaluation function that it managed to harvest a great deal of useful information from this much larger collection. Due to the highly interconnected nature of the case-domain graph and the fact that the neighbor methods can generate many neighbors, it is typical behavior for the system to open a large number of nodes and close far fewer. It is also typical for BankXX with the argument-piece evaluation function to harvest even fewer items than the number of nodes closed due to the fill limits on argument pieces.

The order of the case-domain-graph nodes closed is given in Figure 24:

```
1. [AKIN], 9
2. [ESTUS-THEORY-LEGAL-THEORY], 6
3. [KITCHENS-KULL-THEORY-LEGAL-THEORY], 8
4. [PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY], 8
5. [FLYGARE-THEORY-LEGAL-THEORY], 6
6. [ALI], 6
7. [RIMGALE], 7
8. [OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION-LEGAL-THEORY], 8
9. [GOEB], 7
10. [STUDENT-LOAN-FACTUAL-PROTOTYPE], 6
11. [GUNN], 9
12. [GIBSON], 9
13. [DEANS], 6
14. [IACOVONI], 6
15. [OWENS], 5
16. [AKIN-FACTOR-ANALYSIS], 1
17. [ALI-FACTOR-ANALYSIS], 1
18. [GUNN-FACTOR-ANALYSIS], 1
19. [GIBSON-FACTOR-ANALYSIS], 1
20. [OWENS-FACTOR-ANALYSIS], 1
21. [MAKARCHUK], 0
22. [CRUZ], 0
23. [BAEZ], 0
24. [ASHTON,], 0
25. [BURRELL], 0
26. [RASMUSSEN], 0
27. [CHURA], 0
28. [FLYGARE], 0
29. [SANDERS], 0
30. [ABUSE-OF-CHAPTER-13-LEGAL-THEORY], 6
```

**Figure 24.** Order of case-domain-graph nodes closed when BankXX is run on the *Estus* case as a problem case with the argument-piece evaluation function. The numbers given are the evaluation function values of nodes at the time they are closed.

## 4.5. Comparison with the *Estus* case - Discussion

In order to determine how well BankXX has performed, we compare BankXX output with the cases and legal theories identified in the actual *Estus* opinion (*In re Estus,* 695 F.2d 311 (8th Cir. 1982). See Appendix F. Since certain distinctions are hard to make when encoding an actual court opinion. we combined:

1. APPLICABLE-LEGAL-THEORIES and NEARLY-APPLICABLE-LEGAL-THEORIES,
2. BEST-SUPPORTING-CASES and ordinary SUPPORTING-CASES, and
3. BEST-CONTRARY-CASES and ordinary CONTRARY-CASES.

For the comparison we omitted from the output any cases decided or theories promulgated after *Estus*, a 1982 case,[22] since these are irrelevant to the actual *Estus* case.[23] See Figures 25 and 26. Note that ordinarily dates are ignored in the current version of BankXX since our assumption is that a case put to BankXX will be a new problem case, and thus anything already known to BankXX (i.e., present in the case-domain graph) is potentially relevant and is fair game for consideration by BankXX.

```
AGGREGATED-THEORIES:
ABUSE-OF-CHAPTER-13-LEGAL-THEORY (before 1980)
PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY (before 1980)
ESTUS-THEORY-LEGAL-THEORY (1982)
KITCHENS-KULL-THEORY-LEGAL-THEORY (1981)
OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION-LEGAL-THEORY (before 1980)
FLYGARE-THEORY-LEGAL-THEORY (1983)

LEADING-CITED-CASES:
IACOVONI (1980), DEANS (1982), GOEB (1982), RIMGALE (1982)

AGGREGATED-SUPPORTING-CASES:
GIBSON (1985), GUNN (1984), AKIN (1984)

AGGREGATED-CONTRARY-CASES:
OWENS (1988), ALI (1983), GOEB (1982), RIMGALE (1982)

FACTUAL-PROTOTYPE-STORY:
STUDENT-LOAN-FACTUAL-PROTOTYPE
```

**Figure 25.** Aggregated argument pieces at conclusion of processing on the Estus-problem case. Dates of cases and theories are shown.

[22] We did not eliminate items decided in the same year as *Estus* even though a few, no doubt, were actually decided after the *Estus* case was announced. Many theories existed prior to 1980, the date of our earliest case, and are never deleted.

[23] This approach to dealing with post-dated cases for comparison purposes is not the best since it works against BankXX. This is particularly true for BankXX run with the argument-piece and argument-factor evaluation functions. It is not so much of a burden in the node-type evaluation function. In particular with the argument-piece evaluation function, there are built-in limits on how many items can be harvested for a given argument piece: there is a limit of 3 *supporting cases,* 5 *best suppoorting cases,* 3 *contrary cases*, 3 *best supporting cases*, etc. Post-processing filtering for dates hurts BankXX in two ways: (1) it allows BankXX to use valuable resources to chase down "irrelevant" (i.e., post-dated) cases, and (2) it allows BankXX to fill up on such "irrelevant" cases. A better approach is to check for dates at the time a node is expanded or opened; this is the approach taken in the comprehensive BankXX experiments reported in the companion paper.

```
AGGREGATED-THEORIES:
ABUSE-OF-CHAPTER-13-LEGAL-THEORY
PER-SE-MINIMUM-PAYMENT-REQUIREMENT-LEGAL-THEORY
ESTUS-LEGAL-THEORY
KITCHENS-KULL-LEGAL-THEORY
OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION-LEGAL-THEORY

LEADING-CITED-CASES:
IACOVONI, DEANS, GOEB, RIMGALE

AGGREGATED-SUPPORTING-CASES:


AGGREGATED-CONTRARY-CASES:
GOEB, RIMGALE

FACTUAL-PROTOTYPE-STORY:
STUDENT-LOAN
```

**Figure 26**. Argument piece items for the Estus-problem case that remain after date-filtering. Cases decided or theories promulgated after 1982—the date for *Estus*—have been deleted.

A partial comparison of BankXX output with the hand-coded version of the *Estus* opinion is given in Figure 27. *Overlap* means an item was found by BankXX and the opinion. *Missed* means it was present in the opinion but not harvested by BankXX. *Additional* means it was not present in the opinion but was harvested by BankXX.

```
AGGREGATED-THEORIES:
OVERLAP                  MISSED                                ADDITIONAL
ABUSE-OF-CHAPTER-13      ALL-THE-FACTS-AND-CIRCUMSTANCES       KITCHENS-KULL-THEORY
PER-SE-MINIMUM-PAYMENT-  SUBSTANTIAL-OR-MEANINGFUL-REPAYMENT   OLD-BANKRUPTCY-ACT-
        REQUIREMENT      BEST-INTERESTS-OF-CREDITORS-TEST           GOOD-FAITH-DEF'N
ESTUS-THEORY

LEADING-CITED-CASES:
OVERLAP                  MISSED                              ADDITIONAL
RIMGALE
GOEB
DEANS
IACOVONI

AGGREGATED-SUPPORTING-CASES:
OVERLAP                  MISSED                              ADDITIONAL
                         HEARD
                         IACOVONI*
                         KULL
                         TERRY

AGGREGATED-CONTRARY-CASES:
OVERLAP                  MISSED                              ADDITIONAL
RIMGALE                  DEANS*
GOEB                     BARNES
                         BELLGRAPH
```

**Figure 27.** Partial comparison of the BankXX-generated argument with the hand-coded version of the *Estus* opinion, after argument pieces have been aggregated and post-dated cases removed.
* Note that *Iacovoni* and *Deans* are listed as overlap LEADING-CITED-CASES.

The cases and theories that BankXX has culled from the case-domain graph show considerable overlap with those present in the actual *Estus* opinion. (The full encoding of the *Estus* case is given in Appendix F.) BankXX finds all of the cases in the actual *Estus* opinion that are considered leading cited cases: *Deans*, *Goeb*, *Rimgale*, and *Iacovoni*. *Deans*, in particular, was deemed "persuasive" by the court (opinion at 316). BankXX has identified the student loan story as the prototype story for Estus-problem case. It has output none of the four same-side cases but two of the five contrary cases mentioned in the opinion. Note however that BankXX actually listed the *Iacovoni*, a same-side case, and *Deans*, a contrary case, under leading cases.[24] The cases that BankXX missed are pretty far down in the claim lattice (Figure 15), even when only those cases decided before or in 1982 are considered.

The program has also done quite well in identifying the ABUSE-OF-CHAPTER-13, and the PER-SE-MINIMUM-PAYMENT-REQUIREMENT theories as nearly applicable or applicable to the problem fact situation. Both of these theories were mentioned by the court. It also identified the ESTUS-THEORY, the theory promulgated by this leading case, as nearly applicable to the fact situation. (We noted above that there is not enough information in the Estus-problem case for BankXX to determine whether all the factors defining the ESTUS-THEORY are applicable, and so it harvested only as *nearly-applicable*.) BankXX has also identified some other applicable theories that are not specifically mentioned by the court: the definition of "good faith" under the old bankruptcy statute (The Bankruptcy Act of 1898, ch.541, 30 Stat. 544, as amended), and the KITCHENS-KULL THEORY, promulgated in *Kull* (1981) and reiterated in *Kitchens* (1983). It is also interesting to note that BankXX identified the FLYGARE-THEORY, which doesn't show up in the comparison because of its 1983 date; this theory is really the same as Estus even though it was treated by the Flygare court as its own.

Similar comparisons between the hand-coded *Estus* opinion and aggregated, date-filtered output from BankXX configured with the node-type and argument-factor evaluation functions can also be made. (See Appendix H). There is considerable similarity with what was harvested under the argument-piece evaluation function. However, *Iacovoni* and *Deans* are duly listed respectively in AGGREGATED-SUPPORTING-CASES and AGGREGATED-CONTRARY-CASES and not just in LEADING-CASES; this occurs because BankXX with these other evaluation functions is not restricted by limits on the number of cases that can fill the various argument pieces. In addition, *Heard* and *Barnes*—both missed with the argument piece evaluation function—are harvested in these other runs. *Kull*, *Terry* and *Bellgraph* are still missed.

---

[24]*Iacovoni* is not listed under the AGGREGATED-SUPPORTING-CASES because the relevant same-side argument pieces (SUPPORTING-CASES) were already filled up when *Iacovoni* was considered (i.e., closed) by BankXX and *Iacovoni* is not a <u>best</u> case for the *Estus* problem situation so it was not listed there even though the limit of 5 had not been reached for BEST-SUPPORTING-CASES.. This is an example of how not checking dates during processing can hurt BankXX with regard to comparion purposes: other (post-dated) cases—*Gibson* (1985), *Gunn*(1984), *Akin* (1984)—have already filled up the relevant argument piece and blocked BankXX from including *Iacovoni*. The same is true for *Deans*. Before *Deans* is considered, *Goeb*, *Rimgale* and *Ali* filled the CONTRARY-CASE argument piece, which has a limit of 3 cases. *Deans* is also not a <u>best</u> contrary case. See Figure 22.

Regardless of the evaluation function used, BankXX does not come close to discovering *Bellgraph* since it is never opened. *Bellgraph* is a case that is very meagerly linked with the rest of the case-domain graph and is never opened through expansion by the neighbor methods. On the other hand, *Kull* and *Terry* are opened in all three versions of BankXX. However *Kull* and *Terry* are not leading or best cases and thus are not scored very highly in by the argument-piece evaluation function and in fact, after the *supporting-cases* argument piece is filled (in cycle 12), their value goes to 0. With the node-type evaluation function, *Kull* and *Terry* lose out by the luck of the draw—they are scored 6 along with a host of other cases and simply miss out in tie-breaking. With the argument-factor evaluation function, neither gets a very high score since neither contributes much to the evolving argument as measured by the argument factors.

The same set of theories is retrieved by BankXX with the node-type evaluation function as with the argument-piece evaluation function. BankXX with argument-factor evaluation function retrieves about half as many overall. It misses the important ABUSE-OF-CHAPTER-13-LEGAL-THEORY, as well the same three theories missed by the others. It ends up with only two theories, the PER-SE-MINIMUM theory and the ESTUS-THEORY itself, in common with the *Estus* opinion.

BankXX run with the node-type evaluation function also retrieves the student-loan prototype. This is missed with the argument-factor evaluation function; this is not too much of a surprise since such information is not highly valued with this evaluation function. In general, what is retrieved by each of the evaluation functions reflects their biases, that is, the terms and weights that they use.

In general, BankXX run with the node-type evaluation function **harvests** (i.e., adds to some argument piece) more cases than BankXX run with the argument-piece evaluation function[25] although they both **close** roughly the same number of cases.[26] This is because there are no limits placed on the number of cases that can be harvested for the individual argument pieces when the node-type or argument-factor evaluation functions are used. Limits on the argument pieces when the argument-piece evaluation function is used can be quite restrictive (See Figure 13.) and can have a very significant impact on what is output by BankXX.[27] Thus,

---

[25]For the Estus-problem case, BankXX harvests: 3 AGGREGATED-SAME-SIDE-CASES with the argument-piece evaluation function, 8 with the node-type evaluation function, and 7 with the argument-factor evalaution function. It retrieves 4 AGGREGATED-CONTRARY CASES with the argument-piece evaluation function, 10 with the node-type evaluation function, and 8 with the argument-factor evaluation function. See Figure 25 and Appendix H.

[26]17 cases are closed with argument-piece and 18 with the node-type. With the argument-factor function, 15 are closed.

[27]These limits can greatly impact BankXX's performance under certain circumstances. Notably if the opinion mentions a large number of cases, BankXX will, by definition, miss some. Also, if BankXX harvests post-dated cases (i.e., cases occurring after the date of the problem case), these will not show up in the BankXX-hand-coded comparisons since they are deleted because they are simply not relevant to the problem case. However,

BankXX run with the argument-piece evaluation function isn't just simply BankXX with a different evaluation function but also with bounds placed on what can be harvested by the argument pieces.

Note, that these differences in cases harvested is not immediately apparent by considering the output <u>after</u> it has been date filtered since so many of the harvested cases are post-1982 and thus are deleted. (See Appendix G.) Date-filtering makes BankXX run with the node-type evaluation function appear more conservative and precise than it actually is. BankXX run with the argument-factor evaluation function often harvests fewer cases than BankXX run with either of the other two evaluation functions since the use of this evaluation function is expensive computationally.[28]

It is important to note that even though BankXX's output under the different evaluation functions is similar, it does not behave similarly in its search. The order of exploration of the case-domain graph under the various evaluation functions is quite different,[29] as are the values assigned to individual nodes. Thus, BankXX behaves quite differently with the different evaluation functions even though this might not always be apparent from the output. In a larger case-domain graph, the differences would become more apparent.

There appears to be a classic knowledge-performance trade-off occurring with BankXX run under the various evaluation functions. This is especially evident when non-date-filtered output is examined. BankXX with the node-type evaluation function harvests more cases, including more ADDITIONAL cases not listed in the opinion and fewer MISSED cases, than the BankXX under the other two—especially the argument-piece evaluation function—which have more MISSED but fewer ADDITIONAL cases.[30] This trade-off is persistent. It shows up throughout the extensive set of experiments we have performed on BankXX.

As a rough summary, one can say that BankXX with the node-type evaluation function is somewhat "dumber"—not particularly selective nor sensitive to problem context—whereas BankXX with the argument-piece and argument-factor evaluation functions is "smarter"—more selective and more problem-sensitive. These generalizations are examined in detail in the companion paper.

they have used up a certain amount of BankXX's case limits and possibly prevented relevant, non-post-dated cases from being harvested. A fuller discussion of the problem of evaluation are discussed in the companion article.

[28]Although there are some problem cases, like *Estus*, where BankXX run with the argument-piece evaluation function is very similar to it run with the node-type evaluation function.

[29]Sometimes there are some "chunks" of the case-domain graph that are opened in the same order. This is due to neighbor methods which perform exactly the same under all three evaluation functions.

[30]This is also true if one calculates traditional precision and recall scores for just this one example case using the date-filtered output compared against the hand-encoded opinion. This sort of quantitative analysis is pursued in the companion article.

## Part IV: Related Work and Conclusions

## 5. Related Research

We have not discussed generally here either argument or legal argument, which are treated well and at length elsewhere (e.g., [Perelman & Olbrechts-Tyteca, 1969], [Toulmin, 1958], [Levi, 1949]), or argument modeled through other means than search ([McCarty & Sridharan, 1982], [Sycara, 1989], [Alvarado, 1990]). In addition, our present goal is not to provide a formal, logical model of legal argument. We refer the reader to [Prakken, 1993], [Gordon, 1991] and [Loui *et al*., 1993] for excellent discussions. Recent work by Prakken, Loui and others brings to light some of the connections between a formal analysis of argument and the arguments created by HYPO [Ashley, 1990] and CABARET [Rissland & Skalak, 1991].

Several researchers have addressed aspects of argument as search. [Bhatnagar, 1989; Bhatnagar & Kanal, 1991] treat an argument as a search for a causal model that supports a given proposition. Bhatnagar uses a variant of A* search to create models that satisfy argument goals, in which it assumed that probability values may be computed for the validity of supported propositions given a particular model. While we also view argument creation as theory construction [Rissland & Skalak, 1991], we believe that such a probabilistic approach may be difficult to apply in a domain as "weak" as law.

Branting's GREBE system [Branting, 1991] uses structured representations of the explanations for legal decisions supplied in the opinions of legal cases. The use of factors in BankXX's legal theories is similar to the use of precedent constituents in GREBE. Also, GREBE uses heuristic A* search for one aspect of argument creation: retrieval of a precedent that best explains a problem case. Best-first search is performed in a space consisting of all mappings from a problem case to these structured representations of precedent cases. Thus, GREBE's use of A* search is not in the same search space as that of BankXX, but search is used to the same end—to retrieve relevant cases.

While we do not rely on research using artificial neural networks, or on related, massively parallel techniques, for information retrieval, the flavor of some of this work is similar to ours. In particular, Rose's SCALIR [Rose 1994; Rose & Belew, 1991] is a hybrid symbolic and sub-symbolic system that uses a network of legal knowledge, including Shepard's links and West's key number taxonomy links, through which numerical activation is spread to perform retrieval.

BankXX's approach can be contrasted with SCALIR's approach to legal retrieval. First, while BankXX relies on best-first heuristic search directed by any of three evaluation functions at different levels of abstraction, SCALIR uses spreading activation to perform the retrieval. Thus the search control strategies of the two programs are distinct. Second, approximately 90% of the links in the SCALIR network are weighted connectionist links, with 75% of all the links between cases and terms. BankXX does not incorporate such standard information retrieval term-document indexes, but relies solely on symbolic links between data. SCALIR has no explicit representation for a legal theory or a factual prototype, which are important parts of BankXX's representation. Thus the two semantic networks are quite different. Third, the tasks performed by the programs are different. SCALIR is a generic legal information retrieval program, whereas BankXX's retrieval is directed and informed by the requirements of a particular task: creating an argument. Thus, while SCALIR provides a progressive model for legal retrieval, it is quite different from BankXX in its search control strategy, representation scheme, and task application, as well as on other aspects.

This research also shares certain knowledge representation approaches with earlier work in legal information retrieval (e.g., [Hafner, 1987a, 1987b], [Bing, 1987], [Dick, 1987]). Such projects in conceptual legal retrieval relied on a graph of diverse legal entities and concepts where labeled links captured influences and taxonomic information. A more recent project in legal information retrieval is Gelbart and Smith's FLEXICON [1991], which uses a vector space model for retrieval. FLEXICON can perform automatic thesaurus construction, relevance feedback, and can extract important paragraphs of an opinion to generate headnotes automatically. Our BankXX work also shares certain conclusions on the utility of providing multiple paths to information to aid retrieval demonstrated by earlier work in case-based reasoning (e.g., [Kolodner, 1983]).

Some ideas used in this paper—"in-space" and "cross-space" neighbor methods that make use of graph linkages, interconnected "spaces" of nodes, strongly linked cases and theories, use of indirect "dual space" methods, etc.—echo some of the work first presented on structured representations for (mathematical) knowledge [Rissland, 1977]. For instance, in Rissland's work, there were methods—akin to BankXX boomerang methods—for indexing and retrieving relevant, but not directly or closely linked, items in a given space by visiting nodes in another "dual" space (e.g., the method to "find all the examples that apply the theory used in this example"). As in BankXX, there were also methods that simply followed in-space pointers (e.g., "find all the examples that a particular example references or builds upon"). In fact, the use of examples in that body of work presaged many aspects of current case-based reasoning. However, the structure and methods used by Rissland were much less dynamic than those in BankXX. Most were simply pointer-chasing methods as opposed to those in BankXX, like Type 3 methods, which generate new linkages. In addition, all the indexing in Rissland's work used a static indexing scheme; there was no sense of context-sensitive indexing through dimensions or dynamic neighbor methods. While there were heuristics, there was also no real sense of heuristic search replete with evaluation functions, start nodes, etc.

For the ideas of analyzing representation of cases in terms of important domain factors or "dimensions" [Rissland, et al., 1984], the construction of "claim lattices," which partially order retrieved cases by dimensions, and the selection of best and most on-point cases, we rely on ideas developed in HYPO [Ashley, 1990].

## 6. Conclusions

The goal of this project was to unite a number of areas that bear directly on the process of gathering information for use in legal arguments. The BankXX framework brings together research in information retrieval, control of intelligent computer programs, heuristic search, case-based reasoning, legal research, legal knowledge representation, and, of course, legal argument. In a companion article we discuss a series of experiments designed to measure how well BankXX performs.

This article has described several new approaches and mechanisms including:

- Representation of *legal theories* in terms of domain factors
- *Neighbor methods* for traversing the case-domain graph, a semantic network of case and other legal knowledge from a particular domain
- Three evaluation functions—*node-type, argument-piece, and argument-factor evaluation functions*—to guide search of the case-domain graph, each capturing a different perspective on legal knowledge and argument and incorporating a set of terms to be used in the evaluation
- *Argument pieces* for representing generic information needed for argument
- *Argument factors* for evaluating the quality of an argument

The incorporation of legal theories explicitly into the knowledge representation distinguishes BankXX from previous projects from our group, such as CABARET and HYPO. It also distinguishes BankXX from most other programs in the law. The use of argument pieces and argument factors is also unique to this project.

From the standpoint of case-based reasoning, BankXX has been a testbed to investigate the utility for legal retrieval of applying search in addition to knowledge-based indexing. While HYPO and CABARET retrieved cases indexed by factors and by factors and rules, respectively, BankXX performs retrieval using state-space search in an indexed network of legal knowledge. This application of search effects a data-driven control algorithm for argument creation, with some top-down constraints provided by the need to fill in the argument pieces. The data-driven approach can be contrasted with a top-down control scheme driven by stereotypical argument forms [Skalak & Rissland, 1992].

In addition to describing the computational mechanisms and knowledge representation used in BankXX, we ran through an extended example of BankXX run in a de novo fashion on the *Estus* case, a landmark case addressing the "good faith" requirement that is central for approval of Chapter 13 plans in personal bankruptcy. In addition to illustrating BankXX's overall control flow based on heuristic search, the example presented certain of the computational details, such as the use of neighbor methods, evaluation of opened nodes, and the incremental building up of an argument through the argument pieces as BankXX's problem-solving proceeds.

Even on this single example, certain general themes about BankXX were evident:

- BankXX embodies **strong preferences** for certain kinds of information such as legal theories and leading cases (reified in its evaluation functions).
- BankXX **selectively harvests** useful information from the great wealth of information available for consideration.
- BankXX neighbor methods **greatly expand** the possible leads to examine in the course of problem-solving.
- BankXX's heuristic evaluation (and the fill limits in place when BankXX is configured with the argument-piece evaluation) **greatly limits** the information actually harvested.

The theme of *For many are called, but few are chosen* (Matt 22:14) is particularly evident in the way information is harvested by BankXX configured with the argument-piece evaluation function.

These themes reflect our own personal experience, as well as those of others [Kunz, et al. 1992], in performing legal research in a vast library of traditional legal materials where one is constantly making choices about which information to examine in depth, which leads to follow, which cases and theories to incorporate into one's evolving informational harvest, etc., and all in the context of the need to make an argument, write a brief or memo, possibly on short notice and for a demanding audience. The possibility—nearly reality—of performing such research tasks electronically, possibly with the aid of "infobots" and intelligent network gophers, gives further practical significance to our on work with BankXX.

Although just a single data point, our *Estus* example also illustrates certain qualitative trade-offs between the various configurations of BankXX. Such qualitative and quantitative observations are discussed in the detailed analyses described in the companion paper.

In particular, BankXX configured with the argument-piece evaluation function—and its accompanying fill limits on argument pieces—is much more selective and problem sensitive than BankXX configured with the node-type evaluation function. In fact, without date-filtering BankXX with the node-type evaluation function is not particularly problem-sensitive, at least in terms of what information is ultimately harvested. Of course, the internal behaviors of the various configurations—particularly the order of nodes explored and harvested—do vary greatly. Such differences would no doubt be more apparent in a larger case-domain graph.

As with any exploratory computer program, there were design and implementation decisions that presumably have affected the performance of the program. For example, when the user specifies starting search with a most on-point case, one of the most on-point cases is selected at random and search begins there. The other most on-point cases are discarded in the current implementation, but it would have been simple and probably preferable to place them on the open list as well at the beginning of the search. There is great potential also for identifying prototypical cases, using a family resemblance calculation or some other measure, that was not explored in the current implementation. In retrospect, we would also have increased the limits on the number of items that could fill an argument piece under the argument-piece evaluation. In particular, we would have raised the limits on ordinary *supporting-cases* and *contrary-cases*.

We also learned quite a bit about the impact of dates and date-filtering on performance evaluation. In fact, the issue of dates, while not crucial for the intended use of BankXX as a problem-solving program for new cases is critical in evaluation using already existing cases run as de novo problem cases. These observations lead us to modify BankXX to filter for dates in the course of problem-solving before carrying out the massive set of experiments reported in the companion paper. This change has no impact on our original intentions for BankXX but it does make evaluation more fair.

Analogous considerations regarding court jurisdiction and pedigree probably also exist. However, in our application domain, where there is a paucity of appeals cases and thus, courts tend to look as far afield as they need to for useful precedents, overlooking jurisdiction and pedigree was not a major stumbling block. In other domains, it might.

One area for future work is to learn the evaluation function to do the search (e.g., [Samuel, 1967], [Minton, 1988]). In fact, our inclusion of an evaluation step (using the argument-factors) at the conclusion of a problem-solving run was aimed at this goal. There are a variety of algorithms and architectures that could be applied to learn evaluation functions, such as the fixed-increment error correction rule [Nilsson, 1990], learning from preference predicates [Utgoff & Clouse, 1991], or various neural network algorithms, but most rely on some form of scalar-valued error function to assess the quality of the current evaluation function weights. To apply a technique that relies on linearly ordered supervisory information to evaluate the quality of an argument requires that that quality be expressed in a scalar value. In BankXX, however, the quality of an argument depends on its placement along a variety of argument factors. Thus, at best, the supervisory information available from BankXX is partially ordered and not linearly ordered, unless one combines the argument factors into a scalar value, or finds a learning algorithm that relies on partially ordered fitness values.

Although many legal issues involve the interaction of cases and legal statutes, BankXX does not incorporate statutes or regulations into its domain knowledge. We examined the interaction of arguing with a rule and with cases in detail in the CABARET project [Rissland & Skalak, 1991]. As we noted early in this article, codified legal rules also provide indexes into other types of legal knowledge. The addition of statutory rules to the case-domain graph would also enhance the opportunities for multiple indexing inherent in this domain.

## 7. References

Alvarado, S. J. (1990). *Understanding Editorial Text: A Computer Model of Argument Comprehension*. Boston, MA: Kluwer Academic Publishers.

Ashley, K. D. (1990). *Modeling Legal Argument: Reasoning with Cases and Hypotheticals*. Cambridge, MA: M.I.T. Press.

Ashley, K. D. & Aleven, V. (1991). A Computational Approach to Explaining Case-Based Concepts of Relevance in a Tutorial Context. *Proceedings Case-Based Reasoning Workshop 1991,* 257-268. Washington, D.C. Morgan Kaufmann, San Mateo, CA.

Ashley, K. D. & Rissland, E. L. (1987). But, See, Accord: Generating Blue Book Citations in HYPO. *Proceedings of the First International Conference on AI and Law (ICAIL-87)*, 67-74. Northeastern University, Boston, MA. Association for Computing Machinery.

Ashley, K. D. & Rissland, E. L. (1988). Waiting on Weighting: A Symbolic Least Commitment Approach. *Proceedings Seventh National Conference on Artificial Intelligence (AAAI-88)*, 239-244. Minneapolis, MN. American Association for Artificial Intelligence.

Barletta, R. & Mark, W. (1988). Explanation-Based Indexing of Cases. *Proceedings Case-Based Reasoning Workshop 1988*, 50-60. Clearwater Beach, FL. Morgan Kaufmann.

Barr, A., Feigenbaum, E. A. & Cohen, P. (1981). *The Handbook of Artificial Intelligence.* Reading, MA: Addison-Wesley.

Bareiss, E. R. (1989). *Exemplar-Based Knowledge Acquisition*. Boston, MA: Academic Press.

Berman, D. H. & Hafner, C. D. (1991). Incorporating Procedural Context into a Model of Case-Based Legal Reasoning. *Third International Conference on Artificial Intelligence and Law (ICAIL-91),* pp. 12-20. Oxford, England. Association for Computing Machinery.

Bhatnagar, R. K. (1989). *Construction of Preferred Causal Hypotheses for Reasoning with Uncertain Knowledge*. Ph.D. Dissertation, University of Maryland, College Park, MD.

Bhatnagar, R. & Kanal, L. N. (1991). A Formalism for Automated Generation of Preferred Arguments. *Working Notes, AAAI Spring Symposium Series, Argument and Belief*, 39-61. Palo Alto, CA.

Bing, J. (1987). Designing Text Retrieval Systems for "Conceptual Searching". *The First International Conference on Artificial Intelligence and Law*, pp. 43-51. Boston, MA. ACM.

BlueBook. (1986). *A Uniform System of Citation* (14th ed.). Cambridge, MA: The Harvard Law Review Association.

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

Gardner, A. vdl. (1987). *An Artificial Intelligence Approach to Legal Reasoning*. M.I.T. Press, Cambridge.

Gelbart, D. and Smith, J.C. (1991). Beyond Boolean Search: FLEXICON, A Legal Text-Based Intelligent System. *Third International Conference on Artificial Intelligence and Law (ICAIL-91)*, pp. 225-234. Oxford, England. Association for Computing Machinery.

Hafner, C. D. (1987a). Conceptual Organization of Case Law Knowledge Bases. *Proceedings, First International Conference on Artificial Intelligence and Law*, 35-42. Boston, MA. Association for Computing Machinery.

Hafner, C. D. (1987b). *An Information Retrieval System Based on a Computer Model of Legal Knowledge*. Ph.D. thesis, republished by UMI Research Press, Ann Arbor, MI (1981)., University of Michigan.

Hammond, K. J. (1989). *Case-Based Planning: Viewing Planning as a Memory Task*. Boston, MA: Academic Press.

Knobbe, K. G. (1986). *Hobby and Home Office Deductions — Sections 183 and 280A*. Washington, DC: Tax Management, Inc.

Kolodner, J. L. (1983). Maintaining Organization in a Dynamic Long-Term Memory. *Cognitive Science*, 7(4), 243-280.

Kunz, C. L., Schmedemann, D. A., Bateson, A. L., Downs, M. P. & Erlinder, C. P. (1992). *The Process of Legal Research* (Third Edition ed. **[???]**). Boston, MA: Little, Brown.

Loui, R. P., Norman, J., Olson, J., & Merrill, A. (1993). A Design for Reasoning with Policies, Precedents, and Rationales. *Proceedings of the Fourth International Conference on Artificial Intelligence and Law*, 202-211. Amsterdam, The Netherlands. ACM.

Martin, C. E. (1990). *Direct Memory Access Parsing*. Ph.D. Thesis, Yale University, New Haven, CT.

Minton, S. (1988). *Learning Effective Search Control Knowledge: An Explanation-Based Approach*. Boston, MA: Kluwer Academic Publishers.

Nilsson, N. (1980). *Principles of Artificial Intelligence*. Palo Alto, CA: Tioga Press.

Nilsson, N. J. (1990). *The Mathematical Foundations of Learning Machines* . San Mateo, CA: Morgan Kaufmann.

Owens, C. (1988). Domain-Independent Prototype Cases for Planning. Proceedings, *Case-Based Reasoning Workshop 1988*, 302-311. Clearwater, FL. Morgan Kaufmann, San Mateo, CA.

Perelman, C. & Olbrechts-Tyteca, L. (1969). *The New Rhetoric: A Treatise on Argumentation* . Notre Dame, Indiana: University of Notre Dame Press.

Porter, B. W., Bareiss, R. & Holte, R. C. (1990). Concept Learning and Heuristic Classification in Weak-Theory Domains. *Artificial Intelligence*, 45, 229-263.

Rissland, E. L. (1977). ***Epistemology, Representation, Understanding and Interactive Exploration of Mathematical Theories***. Ph.D. thesis, Department of Mathematics, MIT, 1977.

Rissland, E. L. (1978). Understanding Understanding Mathematics. *Cognitive Science*, 2, 361-383.

Rissland, E. L., Daniels, J. J., Rubinstein, Z. B. & Skalak, D. B. (1993). Case-Based Diagnostic Analysis in a Blackboard Architecture. *Proceedings of the Eleventh National Conference on Artificial Intelligence*, 66-72. Washington, DC. AAAI Press/MIT Press.

Rissland, E. L., Valcarce, E. M. & Ashley, K. D. (1984). Explaining and Arguing with Examples. *Proceedings of the National Conference on Artificial Intelligence. AAAI-84*. Austin, TX. American Association for Artificial Intelligence.

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

Rissland, E. L., Skalak, D. B. & Friedman, M. T. (1993a). Case Retrieval through Multiple Indexing and Heuristic Search. *Proceedings of IJCAI-93*, Chambery, Savoie, France. International Joint Conferences on Artificial Intelligence.

Rissland, E. L., Skalak, D. B. & Friedman, M. T. (1993b). BankXX: A Program to Generate Argument through Case-Based Search. *Proceedings Fourth International Conference on Artificial Intelligence and Law (ICAIL-93),* Amsterdam, The Netherlands.

Rosch, E. & Mervis, C. B. (1975). Family Resemblances: Studies in the Internal Structure of Categories. *Cognitive Psychology*, 7, 573-605.

Rose, D. E. & Belew, R. K. (1991). A Connectionist and Symbolic Hybrid for Improving Legal Research. *International Journal of Man-Machine Studies,* 35, 1-33.

Rose, D. E. (1994). *A Symbolic and Connectionist Approach to Legal Information Retrieval.* Lawrence Erlbaum, Hillsdale, N.J.

Ruby, D. & Kibler, D. (1988). Exploration of Case-Based Problem Solving. *Proceedings Case-Based Reasoning Workshop 1988,* 345-356. Clearwater Beach, FL. Morgan Kaufmann, San Mateo, CA.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM J. Research and Development*, 3:210-229.

Samuel, A. L. (1967). Some studies in machine learning using the game of checkers. II—Recent Progress. *IBM J. Research and Development*, 11:601-617.

Schank, R. C. (1982). *Dynamic Memory,* Cambridge: Cambridge University Press.

Schank, R. C. (1990). *Tell Me a Story: A New Look at Real and Artificial Memory.* New York, NY: Scribner.

Shepard's. (1994). *Shepard's Federal Citations*. Colorado Springs, CO: Shepard's/McGraw-Hill.

Skalak, D. B. & Rissland, E. L. (1992). Arguments and Cases: An Inevitable Intertwining. *Artificial Intelligence and Law: An International Journal*, 1(1), 3-48.

Sullivan, T. A., Warren, E. & Westbrook, J. L. (1989). *As We Forgive Our Debtors* . New York, NY: Oxford University Press.

Sycara, K. P. (1987). *Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods.* Ph.D. Thesis, School of Information and Computer Science, Georgia Institute of Technology. Atlanta, GA.

Sycara, K. P. (1988). Resolving Goal Conflicts via Negotiation. *Proceedings Seventh National Conference on Artificial Intelligence (AAAI-88),* 245-250. Saint Paul, MN. San Mateo, CA: Morgan Kaufmann.

Sycara, K. P. & Navinchandra, D. (1991). Influences: A Thematic Abstraction for Creative Use of Multiple Cases. *Proceedings, Case-Based Reasoning Workshop 1991*, 133-144. Washington, DC. San Mateo, CA: Morgan Kaufmann.

Turner, R. (1988). Organizing and Using Schematic Knowledge for Medical Diagnosis. *Proceedings Case-Based Reasoning Workshop 1988*, 435-446. Clearwater Beach, FL. San Mateo, CA: Morgan Kaufmann.

Utgoff, P. E., & Clouse, J. A. (1991). Two kinds of training information for evaluation function learning. *Proceedings Ninth National Conference on AI (AAAI-91)*, 596-600. Anaheim, CA. MIT Press.

Words and Phrases (1994). *Words and Phrases*. St. Paul, MN: West Publishing.

## Appendix A. The Cases in the BankXX Corpus

| | |
|---|---|
| ADAMU | *In re Adamu*, 82 B.R. 128 (Bkrtcy.D.Or. 1988) |
| AKIN | *Matter of Akin,* 54 B.R. 700 (Bkrtcy. 1985) |
| ALI | *In re Ali*, 33 B.R. 890 (Bkrtcy. 1983) |
| ASHTON | *In re Ashton*, 85 B.R. 766 (Bkrtcy.S.D.Ohio 1988) |
| BAEZ | *In re Baez*, 106 B.R. 16 (Bkrtcy.D.Puerto Rico 1989) |
| BARNES | *Barnes v. Whelan*, 689 F.2d 193 (1982) |
| BELLGRAPH | *Matter of Bellgraph*, 4 B.R. 421 (1980) |
| BOYD | *In re Boyd,* 57 B.R. 410 (Bkrtcy.N.D.Ill. 1983) |
| BROWN | *In re Brown*, 56 B.R. 293 (Bkrtcy.N.D.Ill. 1985) |
| BURRELL | *In re Burrell*, 2 B.R. 650 (1980) |
| CALDWELL | *In re Caldwell*, 895 F.2d 1123 (6th Cir. 1990) |
| CANDA | *In re Canda,* 33 B.R. 75 (Bkrtcy. 1983) |
| CHURA | *In re Chura,* 33 B.R. 558 (Bkrtcy. 1983) |
| CRUZ | *Matter of Cruz*, 75 B.R. 56 (Bkrtcy.D.Puerto Rico 1987) |
| DEANS | *Deans v. O'Donnell,* 692 F.2d 968 (1982) |
| DOS-PASSOS | *In re Dos Passos*, 45 B.R. 240 (Bkrtcy. 1984) |
| EASLEY | *In re Easley,* 72 B.R. 948 (Bkrtcy.M.D.Tenn. 1987) |
| EPPERS | *In re Eppers,* 38 B.R. 301 (Bkrtcy. 1984) |
| ESTUS | *In re Estus*, 695 F.2d 311 (1982) |
| FLYGARE | *Flygare v. Boulden*, 709 F.2d 1344 (1983) |
| GIBSON | *In re Gibson,* 45 B.R. 783 (Bkrtcy. 1985) |
| GIRDAUKAS | *In re Girdaukas*, 92 B.R. 373 (Bkrtcy.E.D.Wis. 1988) |
| GOEB | *In re Goeb,* 675 F.2d 1386 (1982) |
| GUNN | *In re Gunn,* 37 B.R. 432 (Bkrtcy. 1984) |
| HAWKINS | *Matter of Hawkins*, 33 B.R. 908 (Bkrtcy. 1983) |
| HEARD | *In re Heard,* 6 B.R. 876 (1980) |
| IACOVONI | *In re Iacovoni,* 2 B.R. 256 (1980) |
| KITCHENS | *In re Kitchens,* 702 F.2d 885 (1983) |
| KULL | *Matter of Kull,* 12 B.R. 654 (1981) |
| MAKARCHUK | *In re Makarchuk,* 76 B.R. 919 (Bkrtcy.N.D.N.Y. 1987) |
| MARSCH | *In re Marsch*, 11 B.R. 514 (1981) |
| MEMPHIS | *Memphis Bank & Trust Co. v. Whitman,* 692 F.2d 427 (1982) |
| MURALLO | *Matter of Murallo*, 4 B.R. 666 (1980) |
| MYERS | *Matter of Myers,* 52 B.R. 248 (Bkrtcy. 1985) |
| NEUFELD | *Neufeld v. Freeman,* 794 F.2d 149 (4th Cir. 1986) |
| OKOREEH-BAAH | *In re Okoreeh-Baah,* 836 F.2d 1030 (6th Cir. 1988) |
| OWENS | *In re Owens,* 82 B.R. 960 (Bkrtcy.N.D.Ill. 1988) |
| PONANSKI | *In re Ponanski,* 11 B.R. 661 (1981) |
| RASMUSSEN | *In re Rasmussen,* 888 F.2d 703 (10th Cir. 1989) |
| RIMGALE | *In re Rimgale,* 669 F.2d 426 (1982) |
| SANABRIA | *In re Sanabria,* 52 B.R. 75 (D.C. 1985) |
| SANDERS | *In re Sanders,* 28 B.R. 917 (Bkrtcy. 1983) |
| SCHAITZ | *In re Schaitz,* 913 F.2d 452 (7th Cir. 1990) |
| SCHONGALLA | *In re Schongalla*, 4 B.R. 360 (1980) |
| SCHYMA | *In re Schyma,* 68 B.R. 52 (Bkrtcy.D.Minn. 1985) |
| SELLERS | *In re Sellers*, 33 B.R. 854 (Bkrtcy. 1983) |
| SEVERS | *In re Severs*, 28 B.R. 61 (Bkrtcy. 1982) |
| SHEETS | *In re Sheets*, Bkrtcy., 26 B.R. 523 (1983) |
| SILVA | *In re Silva,* 82 B.R. 845 (Bkrtcy.S.D.Ohio 1987) |
| SOTTER | *In re Sotter,* 28 B.R. 201 (Bkrtcy. 1983) |
| STRONG | *Matter of Strong,* Bkrtcy., 26 B.R. 814 (1983) |

| | |
|---|---|
| TAUSCHER | *In re Tauscher,* Bkrtcy., 26 B.R. 99 (1982) |
| TERRY | *In re Terry,* 630 F.2d 634 (1980) |
| TRAMONTO | *In re Tramonto,* Bkrtcy., 23 B.R. 464 (1982) |
| VALENTINE | *In re Valentine,* 29 B.R. 366 (Bkrtcy. 1983) |

## Appendix B. BankXX's Legal Theory Space

The following legal theories are represented in BankXX:

PER-SE-MINIMUM-PAYMENT-REQUIREMENT
LITTLE-INDEPENDENT-MEANING
SUBSTANTIAL-OR-MEANINGFUL-REPAYMENT
ALL-THE-FACTS-AND-CIRCUMSTANCES (ALSO CALLED CASE-BY-CASE-BASIS)
DEANS-THEORY
EASLEY-16-FACTORS
ESTUS-THEORY
FLYGARE-THEORY
ABUSE-OF-CHAPTER-13
OWENS-3-FACTORS
KITCHENS-KULL-THEORY
BEST-INTERESTS-OF-CREDITORS-TEST
MAKARCHUK-PRINCIPAL-PURPOSE-STUDENT-LOAN-DISCHARGE
JOHNSON-ANALYSIS-DISCHARGE-STUDENT-LOANS
OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEFINITION
RIMGALE-THEORY
MEMPHIS-THEORY
OKOREEH-BAAH-THEORY

The following theory links are represented in BankXX's Legal Theory Space:

ALL-THE-FACTS-AND-CIRCUMSTANCES—***agrees-with***—FLYGARE
CH-13-ABUSE—***overlaps-with***—FACTS-AND-CIRCUMSTANCES
ESTUS—***conflicts-with***—MAKARCHUK
ESTUS—***derives***—MAKARCHUK
ESTUS—***overlaps-with***—-WITH-KITCHENS
ESTUS—***rejects***—PER-SE-MINIMUM
ESTUS-THEORY—***is-equivalent-to***—FLYGARE-THEORY
FLYGARE—***agrees-with***—ALL-THE-FACTS-AND-CIRCUMSTANCES
FLYGARE—***rejects***—PER-SE-MINIMUM
FLYGARE-THEORY—***is-equivalent-to***—ESTUS-THEORY
KITCHENS—***overlaps-with***—-ESTUS
LITTLE-INDEPENDENT-MEANING—***rejects***—ESTUS-THEORY
LITTLE-INDEPENDENT-MEANING—***rejects***—SUBSTANTIAL-REPAYMENT
MAKARCHUK—***conflicts-with***—ESTUS
MAKARCHUK—***derived-from***—ESTUS
RIMGALE—***overlaps-with***—ESTUS
RIMGALE—***overlaps-with***—FACTS-AND-CIRCUMSTANCES
RIMGALE—***overlaps-with***—-OLD-BANKRUPTCY-ACT
RIMGALE—***rejects***—PER-SE-MINIMUM
SUBSTANTIAL-REPAYMENT—***conflicts-with***—-ESTUS

## Appendix C. BankXX Domain Factors

The following domain factors are employed in BankXX:

*financial-situation-factor*
*nature-of-unsecured-claims-factor*
*amount-of-unsecured-claims-factor*
*percent-surplus-of-income-factor*
*employment-history-factor*
*earnings-potential-factor*
*likelihood-income-increase-factor*
*plan-duration-factor*
*plan-accuracy-factor*
*inaccuracies-to-mislead-factor*
*preferential-creditor-treatment-factor*
*secured-claims-modified-factor*
*debt-type-factor*
*nondischarge-7-factor*
*special-circumstances-factor*
*frequency-relief-sought-factor*
*motivation-sincerity-factor*
*trustee-burden-factor*
*relative-total-payment-amount-factor*
*relative-monthly-payment-amount-factor*
*use-of-skills-gained-factor*
*relative-educational-loan-debt-factor*
*de-minimis-payments-factor*
*other-relevant-considerations-factor*
*attempts-to-pay-factor*
*repayment-unsecured-debt-factor*
*necessary-expenses-minus-plan-payments-factor*
*unfair-manipulation-factor*
*substantial-repayment-factor*

## Appendix D. BankXX Story Prototypes

The following story prototypes are used in BankXX:

- student-loan
- civil-judgment-lien
- family-farm
- dishonest-debtor
- medical-calamity
- consumer-debt
- automobile-debt
- honest-debtor
- bankruptcy-repeater
- flatbroke
- divorce
- interrupted-income
- desperate-economic-trouble-unrealistic-plan
- entrepreneur
- irresponsible-debtor
- widow
- slimy-middle-class-manipulator
- homeowner

## Appendix E. Description of family resemblance calculation

Family resemblance was calculated following the description provided by Rosch and Mervis, (1975). Given a set of cases to be considered a "family," and a case whose family resemblance is to be computed, for each attribute value appearing in the case, the number of cases in the family that *share* that attribute is counted. The family resemblance of the case is the sum of this count over all the attributes in the case. Complications arise where an attribute is not a single symbolic value that can be matched exactly in order to determine if two cases *share* an attribute. In BankXX's calculation, the user can set a tolerance threshold for numerical values, so as not to require an exact match. By default, the tolerance is set at 20%, so that if a numerical value (such as the length of the proposed plan in months) for a case is within 20% of the value for the case whose family resemblance is being computed, the cases are considered to share that attribute. Where attributes values are lists, the values are considered to match if the two lists are not disjoint, which is reasonable for the short lists found as attribute values in BankXX.

## Appendix F. The Hand-Coded *Estus* Opinion

The following is the hand-coded representation of case and theory information found in the actual opinion of the Estus (*In re Estus*, 695 F.2d 311 (8th Cir. 1982)).

```
AGGREGATED THEORIES:
ALL-THE-FACTS-AND-CIRCUMSTANCES
ABUSE-OF-CHAPTER-13
BEST-INTERESTS-OF-CREDITORS-TEST
PER-SE-MINIMUM-PAYMENT-REQUIREMENT
SUBSTANTIAL-OR-MEANINGFUL-REPAYMENT
ESTUS-THEORY

LEADING-CITED-CASES:
RIMGALE, GOEB, DEANS, IACOVONI

AGGREGATED-SUPPORTING-CASES:
HEARD, IACOVONI, KULL, TERRY

AGGREGATED-CONTRARY-CASES:
RIMGALE, GOEB, DEANS, BARNES, BELLGRAPH,

FACTUAL-PROTOTYPE-STORY:
STUDENT-LOAN
```

## Appendix G. The Estus Problem Case under Node-Type and Argument-Factor Evaluation Functions

The following is aggregated partial output of BankXX run on the Estus-problem case with the node-type evaluation function. Post-1982 items that would be deleted in the post-processing date-filtering are also shown.

```
AGGREGATED-THEORIES:                          DELETED:
ABUSE-OF-CHAPTER-13 (before 1980)             FLYGARE-THEORY (1983)
PER-SE-MIN-PAYMENT-REQUIREMENT (before 1980)
ESTUS-THEORY (1982)
KITCHENS-KULL-THEORY (1981)
OLD-BANKRUPTCY-ACT-GOOD-FAITH-DEF'N (before 1980)

LEADING-CITED-CASES:
RIMGALE (1982),GOEB (1982), DEANS (1982), IACOVONI (1980)


AGGREGATED-SUPPORTING-CASES:                  DELETED:
BURRELL (1980), IACOVONI (1980),              AKIN (1985), RASMUSSEN (1989),
HEARD (1980)                                  SANDERS (1983), CHURA (1983),
                                              MAKARCHUK (1987)

AGGREGATED-CONTRARY-CASES:                    DELETED:
DEANS (1982), BARNES (1982),                  ALI (1983), CRUZ (1987),
GOEB (1982), RIMGALE (1982)                   BAEZ (1989), ASHTON (1988),
                                              SCHYMA (1985), FLYGARE (1983)
```

The following is aggregated partial output of BankXX run on the Estus-problem case with the argument-factor evaluation function. Post-1982 items that would be deleted in the post-processing date-filtering are also shown.

```
AGGREGATED-THEORIES:                          DELETED:
PER-SE-MIN-PAYMENT-REQUIREMENT (before 1980)  FLYGARE-THEORY (1983)
ESTUS-THEORY (1982)
KITCHENS-KULL-THEORY (1981)


LEADING-CITED-CASES:
RIMGALE (1982),GOEB (1982), DEANS (1982), IACOVONI (1980)


AGGREGATED-SUPPORTING-CASES:                  DELETED:
BURRELL (1980), IACOVONI (1980),              AKIN (1985), RASMUSSEN (1989),
HEARD (1980)                                  CHURA (1983), MAKARCHUK (1987)


AGGREGATED-CONTRARY-CASES:                    DELETED:
DEANS (1982), BARNES (1982),                  ALI (1983), CRUZ (1987),
GOEB (1982), RIMGALE (1982)                   BAEZ (1989), ASHTON (1988)
```