Meal Planning from an Abstraction Hierarchy of Menus and Recipes

Adapting Research from AI Categorization and Problem Solving

Douglas H. Fisher

Vanderbilt University
Douglas.h.fisher@vanderbilt.edu

Abstract. This paper describes a theoretical framework for a system for menu planning and recipe design, which builds on a previously developed AI system for problem solving that proceeds, in large part, through categorization. The task addressed in this framework is the retrieval and creation of recipes and menus that are consistent with stated constraints, and that are ordered in their presentation by factors such as projected value, novelty, and unexpectedness.

Keywords. Recipe, meal, menu, complementarity, abstraction hierarchy, problem-solving, categorization, AND-OR search, creativity

1 Introduction

This paper describes how a library of recipes, meals, and menus can be used for reuse, reconfiguration, and remixing by human and/or AI chefs and cooks. This work is building on research from machine learning and AI on problem solving through categorization (in the domain of math problem solving), as well as prior research on AI for cooking (e.g., Müller & Bergmann, 2014, 2015; Gaillard, Lieber, & Nauer, 2015). The goal is to extend this framework to the weaker domain theory problem of recipe design, by using a library of recipes and menus that are stored in an abstraction hierarchy. These conceptual underpinnings are related to case-based approaches (e.g., Muñoz-Avila & Cox, 2008; Hammond, 1990), a link that was recognized early (e.g., Fisher & Yoo, 1993, Section 4.3). *A system is, as yet, unimplemented.*

The particular task addressed in this framework is the identification of recipes and menus through exploitation and exploration processes of retrieval and creation, respectively. Recipes are consistent with stated constraints, and recipes and menus can be ranked by projected value (e.g., taste, nutrition), novelty, and unexpectedness.

2 AI Categorization and Problem Solving

Categorization and problem solving research that is relevant to culinary creation stems from a knowledge-rich form of machine learning called explanation-based learning (Flann & Dietterich, 1989), with the goal of speeding up deductive reasoning

processes like theorem proving, by remembering previously discovered packages of composite knowledge (e.g., theorem proof traces or other "explanations") in a repository, and reusing knowledge from this repository to speed up subsequent deductive problem solving. Even though ***speed-up is not our goal in culinary creativity***, there is nonetheless utility in the explanation-based paradigm for our purposes.

A limitation of the explanation-based approach is that it requires a domain theory of inference rules, which are often assumed to be "perfect". Thus, explanation-based reasoning is often regarded as deductive. But the same explanation-based machinery can use inference rules that do not yield deductively-sound conclusions, so long as those rules are formatted consistent with system specification. So while the need for a domain theory is a legitimate limitation, the domain theory can be inconsistent, which is particularly important in the recipe design setting.

2.1 Induction over explanations and problem solutions

Yoo and Fisher (1991) and Fisher and Yoo (1993), in their system EXOR (Explanation Organizer), combined explanation-based learning with more traditional forms of inductive machine learning, such as generalization (Flann & Dietterich, 1989), so as to hierarchically structure and index the derived knowledge repository through unsupervised, inductive clustering. The hierarchical knowledge base constructed in this way could be navigated more effectively when searching for knowledge that was most relevant to a current problem. In the framework for case-based problem solving (e.g., Muñoz-Avila & Cox, 2008), generalization is a type of proactive case merging.

The effect of EXOR is to restructure the AND-OR search that is common in problem solving, from a search space in which OR and AND nodes are interleaved and alternating, to a single OR tree of partial explanations (aka problem solving traces). In turn, each explanation (aka a node in the OR tree) is an AND tree that represents a (partial) solution. Partial explanations accumulate details as the OR tree is traversed from root to leaf, where the root is the empty design (aka explanation or solution) and a leaf is a fully-realized design in the form of a complete AND tree (though EXOR uses pruning mechanisms, so that leaves are often only partial solutions or designs).

This ***single OR tree (of AND trees)*** has the advantage that it can be more easily indexed for purposes of speeding traversal in search of past knowledge that is most relevant to a current problem. Indices that correspond to features of the problem statement (aka surface features), or features that are inferred from those features (aka deep features) are used to direct the categorization process based on a best match between features of the problem being solved and solutions in the hierarchy.

In EXOR, cost effective features for indexing are identified that optimize a tradeoff between the cost of inferring the presence of the feature in the current problem, and the search through the space of AND tree designs that is saved by using the feature as an index into the OR tree. These indexing features are EXOR's implementation of a boundary of operationality (Bravermann & Russell, 1988) as cost-effective features, which may be overtly observed surface features, as well as inferred deep features.

2.2 Solving new problems by categorization with domain theory search

With the abstraction hierarchy (OR tree) over problem solving traces at varying levels of abstraction, and indices to guide search, EXOR solves a new problem by traversing the abstraction hierarchy in a meaningful way, guided by features of the problem, "accumulating" a single complete solution as it does so. However, traversing an EXOR tree is still an AND/OR search, but one that is typically more efficient than a search is a less structured space. Unlike categorization in other paradigms, such as classification with a decision tree, categorization in an EXOR-induced tree is not strictly deterministic, but rather backtracking happens if and when there is a contradiction between what is known about a current problem and a partial solution.

Importantly, if there is no current complete solution in the EXOR tree that solves a problem without conflicts, then EXOR reverts to searching background knowledge for a way to complete an existing partial solution. Thus, EXOR can expand its hierarchy of solution traces as needed, through a process of categorization-centric problem solving, so long as it has a complete (even if inconsistent) domain theory. In case-based terms, the role of cases can be thought of as providing search control knowledge (Muñoz-Avila & Cox, 2008), albeit mediated by generalizations.

3 Adapting the EXOR Framework to Culinary Creation

Instead of AND-tree-structured problem solutions in a domain of algebra story problems, which was EXOR's original test domain, this paper describes a proposed adaptation of EXOR that organizes and creates AND-tree-structured culinary constructs, with annotations in these constructs that represent assemblage processes of mixing, cooking, and presentation.

3.1 Some related work in the cooking domain

In the cooking domain particularly, EXOR can be viewed as combining aspects of compositional and generative adaptation used by other culinary assistants (e.g., Müller & Bergmann, 2014, 2015). The compositional aspect occurs as categorization, with potential backtracking, pieces together a solution trace. The generative aspect occurs when there is an appeal to domain theory search. Müller & Bergmann also use feature similarity to guide the search for best matching case structures (i.e., workflows), with inference rules in the form of an isa-hierarchy allowing for both surface and deep features to participate in similarity computations. While Müller & Bergmann allows a limited form of parameter generalization, they also elaborate workflow *streams* and *streamlets*, which can be viewed as forms of structural generalization.

Research by Gaillard, Lieber, & Nauer (2015) also uses an isa-hierarchy of ingredients to guide search for similar cases, as well as a concept lattice, which is a graph generalization of an EXOR-like abstraction hierarchy.

If there is a niche for an EXOR-like recipe assistant relative to these and other efforts, it is perhaps in the form of abstractions, and selectivity with which EXOR creates and maintains abstractions. It also is the case that EXOR predates much of the

other work, and appears to still have novel contributions to make on generalizing cases, cost-effective indexing of cases, and "optimal" levels of generalization, in the cooking domain and otherwise. This last concept of "optimal" levels of generalization, or basic levels, supposes that there is an abstraction level of problem solving that provides the "most bang for the buck" (Fisher & Yoo, 1993). Presumably, this extends to the cooking domain as well, though problem solving speed may be less dominant a concern in the cooking domain, as in problem solving creativity.

3.2 Representing context, ingredient, and process knowledge

A portion of a potentially very large hierarchy of recipes is shown in figure 1. The paths in the hierarchy (OR tree) that are enumerated from root to leaves show how each child of a node (an AND tree) expand the AND tree of its parent node, and how different children of a common parent expand the parent in different ways. In addition to the abstraction over ingredient-based AND-trees that are depicted in the figure, there can be annotations of each AND tree that reflect processing guidelines for preparing recipes by mixing, cooking, and presenting. These annotations indicate processes that could be carried out by multiple agents (e.g., in a restaurant kitchen), or just one agent, with asynchronous processing allowed, but under some synchronizing constraints (e.g., "mix the dry ingredients" presumably in any order, then "stir in the eggs and milk"). In any case, we desire a representation of recipes that include

- the ingredient-based AND-trees;
- the varying quantities on ingredients;
- context information, such as who commissioned a meal; and
- the recipe's asynchronous procedural guidelines for assembling dishes (e.g., mixing ingredients and cooking), under constraints.

The inclusion of context features, such as the "Chancellor commissioned the meal" and the inferences that follow from that, may be a novel functionality.

As noted earlier, the ingredient-based AND trees are similar to nodes of a concept lattice used by Gaillard, Lieber, & Nauer (2015).

The workflow representations of Müller and Bergmann (2014, 2015) are an existing formalism for representing the process knowledge in cooking, as well as the ingredient knowledge. At a minimum, it will be an interesting exercise to adapt the workflow representation to EXOR-style hierarchies, which appear to generalize the data node constructs in Müller and Bergmann.

The TAEMS framework (Decker & Lesser, 1993; Szekely, et al, 2006) for Task Analysis, Environment Modeling, and Simulation, is an alternative. TAEMS is promising in its application to multi-agent contexts, which can be particularly important for representing a kitchen environment with collaborating cooks. Using either the TAEMS or workflow formalisms, EXOR-style AND trees within each node would be augmented with links that indicate any required ordering on operations in the recipe – for example, that dry ingredients need to be mixed before adding wet ingredients; that the oven should be preheated; that a cake should be cooled before applying frosting.

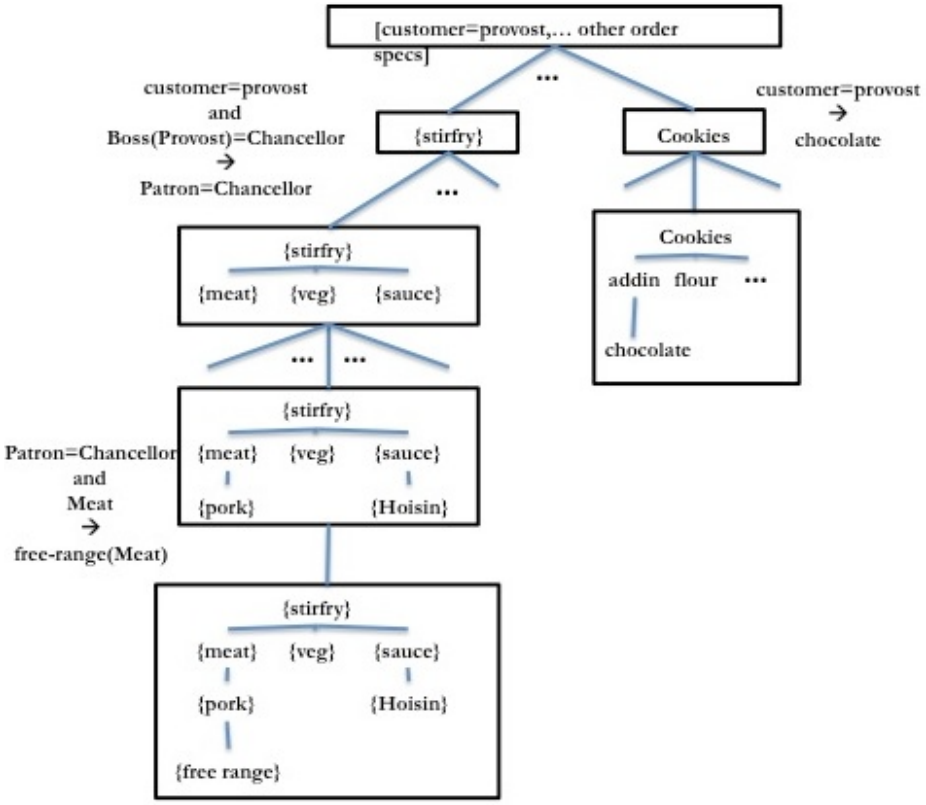

Fig. 1 gives an abstraction hierarchy of recipes. The root shows the initial constraints and context for a dish to be created, such as the nature of the event (from which other conditions can be inferred through domain theory like the need for free range meat), desired or required ingredients (e.g., Hoisin sauce) or styles (e.g., low fat, stir fry), and forbidden ingredients (e.g., nuts).

3.3 Goal-directed recipe creation

A recipe creation agent can use an augmented abstraction hierarchy of recipes in a variety of ways. If a set of constraints is specified, such as a fixed set of required ingredients (e.g., think the TV show, Chopped), much like a problem statement in EXOR's prior life, then the recipe creation agent can act much like EXOR, through categorization-centric problem solving. The goal in the cooking setting is any recipe or menu that satisfies the constraints (e.g., set of required ingredients). Of course, many recipes may satisfy the goal conditions, so it is desirable to rank minimally-satisfactory recipes by factors such as value (e.g., taste, nutrition, carbon footprint), novelty, and unexpectedness.

To review, the recipe creation agent would find a path in the hierarchy that is suggested by cost-effective indexing features, but which also satisfies the constraints (e.g., of required ingredients), and reverting to less-structured domain knowledge, as

necessary, to complete partial recipes found through categorization. Domain knowledge in this setting would include transformation operators, like those specifying that one ingredient (e.g., apple sauce) might be a good substitute for another (e.g., butter) under particular circumstances (e.g., baking), and that one cooking method can be substituted for another (e.g., Gaillard, Lieber, & Nauer, 2015; Muñoz-Avila & Cox, 2008; Hammond, 1990).

Constraints other than requisite ingredients that can be used to prune the search of the recipe hierarchy include ingredients to be excluded, such as refined sugar, gluten, tree nuts, and other health-motivated exclusions. In the culinary domain, cost effective features that might direct categorization could be ingredients that complement ingredients in a partial recipe or the identity of a special patron for which the recipe is being created. Figure 1 illustrates that chocolate is introduced as an ingredient at one node because of a known customer preference that is encoded as an inference rule. Elsewhere in the hierarchy, "patron=Chancellor" is inferred due to another rule, and this feature will bias (or hard constrain) categorization down some sub-trees and not others. The nature of the domain theory can vary from generalized knowledge (e.g., about frequent ingredient substitutions), to highly specific context knowledge about the individual patron preferences and restrictions.

More generally, instead of searching the abstraction hierarchy for a single recipe solution, search can continue to find multiple solutions, perhaps for anytime motivations (e.g., again, think Chopped), and/or to uncover a number candidate recipes that can be assessed in terms of novelty, utility, and surprise relative to the recipes that are in the existing recipe repository (Maher & Fisher, 2012; Grace, et al, 2014).

3.4 An important and interesting aside: meal and menu planning

Traversing the recipe hierarchy can be extended to search for multiple recipes that are complementary, which is important in meal and menu creation. A restaurant or catered *meal* is composed of a combination of dishes that are complementary, where each dish is created through a recipe. A meal shares the desire for complementarity among its components with a recipe, albeit with more loosely-coupled procedures for the assemblage of the former than the latter. A *menu* for a catered buffet also includes several-to-many dishes, each of which is again created through a recipe. While there is attention to complementarity in a buffet menu, there is greater flexibility in how patrons sample and assemble their individual meals, and so the complementarity (or coupling) between any pairs (or combinations more generally) of particular dishes is less. The complementarity between meals in a non-buffet restaurant menu is more loosely coupled still, driven more by resources of the kitchen than customers, very few of whom would order more than one meal at a sitting.

In sum, dishes/recipes are composed of atomic foods and/or composite sub-recipes; meals are composed of dishes/recipes; and menus are composed of dishes as well, with or without intervening meal constructs. It is tempting to treat meal and menu planning as special cases of "recipes", but with differing rules and guidelines for composition – ones that reflect different demands for complementarity, both of taste and nutrition. To some extent, this theoretical union may be productive.

3.5 Increased creativity in recipe creation through increased exploration

Specificity of recipes provides constraints, as do (low-level) component composite foods (e.g., an Alfredo sauce) that are available for use in larger recipes. These lowest level composite constructs may correspond to *streamlets* (Müller and Bergmann (2014, 2015). Between high-level recipe template and low-level components, are mid-level recipe-design elements, which if fully fleshed out, give no room for creativity, and if not sufficiently fleshed out, recipe design may default to design from scratch. An issue that should be addressed is related to the previously raised idea of boundaries of operationally, but rather than being concerned with increasing the likelihood of beneficial reuse for problem solving speed up, we want optimal (or basic) levels of abstraction for purposes of creative recipe design, with a particular eye towards designs that are novel, utile, and surprising.

In terms of the well-known and pervasive *exploitation-exploration tradeoff*, EXOR was heavily biased towards exploitation. However, rather than resorting to domain theory search only when fully-specified recipes fail to satisfy all constraints, in the recipe creation agent, a relaxed version of categorization-centric problem solving would appeal to domain theory search at intermediate nodes, more proactively by deviating from established paths.

An important issue in this regard is the criteria that the agent should use to explore by deviating into a less constrained domain theory search even before it has exhausted all options to exploit. If we were still in the classic problem solving realm, our temptation to venture away from established chunks or macros (e.g., Iba, 1989), would be when the domain theory search was likely to be most narrow (i.e., most constrained), but in creative design, the criteria will be multi-faceted, and opportunities for deviating away from the straight and narrow are an important factor.

4 Concluding Remarks

The paper describes computational cognitive strategies that underlie a potential recipe creation agent, based on earlier work on categorization and problem solving, and thanks to reviewers, by prior research on AI cooking assistants. While this work is clearly related to case-based approaches that seem dominant in the cooking domain, the work on generalization towards *basic levels of problem solving* (Fisher & Yoo, 1993) seems novel relative to CBR generally, and in the cooking domain particularly.

There are very interesting issues that are currently underdeveloped, which would be desirable to talk through with workshop participants. How can ingredient AND-trees be best augmented with the procedural information (e.g. with TAEMS) that are an important aspect of recipes, be folded into the generalization operations of an EXOR-like recipe creation agent? What are ideal criteria for managing the exploration and exploitation tradeoff in search for creative recipes, meals, and menus? What is the potential advantage of proactive case merging – aka generalization – to improve (or not) recipe design?

5 Acknowledgements

Three very helpful reviews pointed to a substantial body of related work.

6 References

1. Braverman, M., & Russell, S. (1988). Boundaries of Operationality. Proceedings of the Fifth International Conference on Machine Learning (pp. 221-234). Ann Arbor, MI: Morgan Kaufmann, 1988.
2. Decker, K., & Lesser, V. 1993. Quantitative modeling of complex computational task environments. In Proceedings of the 11th National Conference on Artificial Intelligence (AAAI-93), 217–224.
3. Fisher, D., & Yoo, J. (1993). Categorization, Concept Learning, and Problem Solving: A Unifying View. In G. Nakamura, R. Taraban, and D. Medin (Eds). The Psychology of Learning and Motivation, 29, San Diego, CA: Academic Press, pp. 219-255. Retrieved from http://www.vuse.vanderbilt.edu/~dfisher/Papers/EmpiricalAndAnalyticHybrids/PsycLearnMotiv93.pdf.
4. Flann, N. S., & Dietterich, T. G. (1989). The Study of Explanation-Based Methods for Inductive Learning. Machine Learning, 4, 1987-226.
5. Gaillard, E., Lieber, J., & Nauer, E. (2015). Improving Ingredient Substitution using Formal Concept Analysis and Adaptation of Ingredient Quantities with Mixed Linear Optimization. In Proceedings of the ICCBR 2015 Workshops. Frankfurt, Germany.
6. Grace, K., Maher, M., Fisher, D., & Brady, K. (2014). A data-intensive approach to predicting creative designs based on novelty, value and surprise. International Journal of Design Creativity and Innovation.
7. Hammond, K. (1990). Explaining and repairing plans that fail. Artificial Intelligence, Vol. 45, pp. 173-228.
8. Iba, G. (1989). A Hueristic Approach to the Discovery of Macro-Operators, Machine Learning, Vol. 3, pp. 285-318.
9. Maher, M. L., & Fisher, D. (2012). Using AI to Evaluate Creative Designs. In The 2nd International Conference on Design Creativity. Glasgow, UK.
10. Müller, G., & Bergmann, R. (2014). Compositional Adaptation of Cooking Recipes using Workflow Streams. In Computer Cooking Contest, Workshop Proceedings ICCBR 2014, Springer, 2014.
11. Müller, G., & Bergmann, R. (2015). CookingCAKE: A Framework for adaptation of cooking recipes represented as workflows. In Computer Cooking Contest, Workshop Proceedings ICCBR 2015, Springer, 2014.
12. Muñoz-Avila, H., & Cox, M. T (2008) Case-Based Plan Adaptation: An Analysis and Review, IEEE Intelligent Systems, Vol. 23, 4, pp. 75-81.
13. Szekely, P., Maheswaran, R. T., Neches, R., Rogers, C. M., Sanchez, R., Becker, M., Fitzpatrick, S., Gati, G., Hanak, D., Karsai, G., & van Buskirk, C. (2006). An Examination of Criticality-Sensitive Approaches to Coordination, In AAAI Spring Symposium on Distributed Plan and Schedule Management, pages 136--142, 2006. AAAI Press. Retrieved from http://www.isi.edu/~szekely/contents/papers/2006/SS0604SzekelyP.pdf.
14. Yoo, J., & Fisher, D., (1991). Concept Formation over Explanations and Problem Solving Experience. In Proceedings of the 12th International Joint Conference on Artificial Intelligence (pp. 630-636). Sydney, Australia. Morgan Kaufmann.