Soup Over Bean of Pure Joy: Culinary Ruminations of an Artificial Chef

Richard G. Morris, Scott H. Burton, Paul M. Bodily, and Dan Ventura
Computer Science Department
Brigham Young University
rmorris@axon.cs.byu.edu, sburton@byu.edu, norkish@gmail.com, ventura@cs.byu.edu

Abstract

We introduce a system for generating novel recipes and use that context to examine some current theoretical ideas for computational creativity. Specifically, we have found that the notion of a single inspiring set can be generalized into two separate sets used for generation and evaluation, respectively, with the result being greater novelty as well as system flexibility (and the potential for natural meta-level creativity), that explicitly measuring artefact typicality is not always necessary, and that explicitly defining an artefact hierarchically results in greater novelty.

1 Introduction

As a relatively new sub-field of artificial intelligence (AI), computational creativity is currently wrestling with many issues similar to those with which AI struggled several decades ago. Many questions similar to those originally asked of AI are now being asked in the context of computational creativity, including foundational questions such as "What is creativity?" Within computational creativity, there is an ongoing movement to define a theoretical foundation that can provide a level of maturity to the field. For example, Wiggins gives the following definition of computational creativity that closely mirrors definitions of intelligence accepted by many AI researchers (Wiggins 2006):

> *The study and support, through computational means and methods, of behaviour exhibited by natural and artificial systems, which would be deemed creative if exhibited by humans.*

As another example, Ritchie provides a level of formalism by supplying a framework for evaluating a creative system (Ritchie 2007). Assuming that a creative system's purpose is to produce creative artefacts, Ritchie's framework evaluates the creativity of the system in terms of the typicality and quality of generated artefacts in relation to some inspiring set of known artefacts.

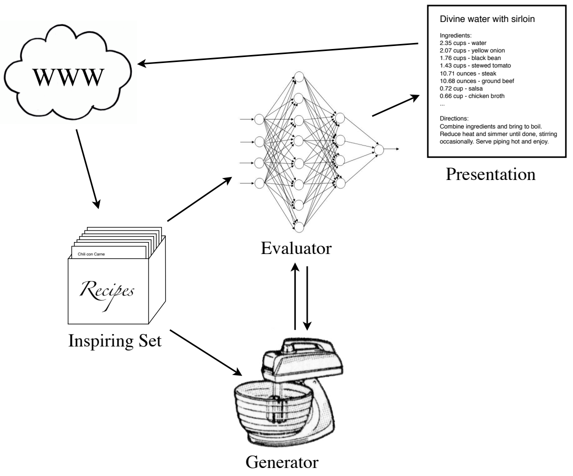
Taking some of Ritchie's ideas one step further, Gervás proposes that creative systems must be able to consistently generate creative artefacts—producing artefacts that are also novel with respect to its own previous work (Gervás 2011). Gervás shows this can be accomplished by splitting the inspiring set (as discussed by Ritchie) into a reference set (used to determine the novelty of generated artefacts) and a learning set (used in the generation of artefacts). We modify this idea by splitting the inspiring set into a set used in the generation of artefacts and one for evaluating generated artefact quality. Note that this does not address the idea of a reference set at all, but it also does not preclude the use of one either (let us say the two ideas are orthogonal and likely complementary).

Evaluation of a creative system is both clearly important and inherently difficult. In a recent comprehensive survey of published creative systems, Jordanous found that only half of the papers give details on an evaluation of their system (Jordanous 2011). Despite the difficulty in measuring creativity, quality, and typicality, greater attempts must be made to evaluate them if the field is to gain maturity.

In an attempt to do so, we provide an explicit measure of quality used during the artefact generation process. We also show that an explicit measure of typicality is not necessary if it is built in to the generation process. In addition, we present an explicit measure of novelty (rare *n*-grams). We also show that explicitly defining a hierarchy for elements of our artefacts is beneficial to the creative system. We compare a hierarchical version of our system with one that is lacking any hierarchy and demonstrate greater novelty in the artefacts produced. The hierarchical version also gives a natural method to implicitly model typicality in the system without inhibiting novelty.

Novel perspectives on the developing theory of computational creativity are provided by concrete applications of the theory in diverse areas. Creative systems have been produced for a wide variety of artefacts, including poetry (Gervás 2000; Gervás 2001), literature (Pérez y Pérez and Sharples 2001; Pérez y Pérez 2007), music (Jordanous 2010; Lewis 2000; Monteith et al. 2011), theorem proving (Ritchie and Hanna 1984; Colton 2002), humor (Stock and Strapparava 2005; Binsted and Ritchie 1997), metaphor (Veale and Hao 2007), and art (Cohen 1999; Colton 2008; Norton, Heath, and Ventura 2011). The distinctive context of each of these concrete applications provides a novel perspective on the developing field of computational creativity. Further exploration of new domains provides additional viewpoints to help the theory mature. To this end, we present a creative system for recipe generation.

While work on recipes has been done in the field of artificial intelligence, to our knowledge, a recipe generation sys-



Divine water with sirloin

Ingredients:
2.35 cups - water
2.07 cups - yellow onion
1.76 cups - black bean
1.43 cups - stewed tomato
10.71 ounces - steak
10.68 ounces - ground beef
0.72 cup - salsa
0.66 cup - chicken broth
...

Directions:
Combine ingredients and bring to boil. Reduce heat and simmer until done, stirring occasionally. Serve piping hot and enjoy.

Figure 1: High-level view of the system architecture. Inspiring set recipes are taken from online sources and inform the evaluator and generator. Recipes are created through an iterative process involving both generation and evaluation. Eventually, generated recipes with the highest evaluation are fed to the presentation module for rendering and may be published online.

tem whose focus is *creativity* has not yet been developed (or even attempted). These other AI recipe generators use case-based reasoning to plan out a recipe, in the case of CHEF (Hammond 1986), or a meal, in the case of Julia (Hinrichs 1992). These approaches maximize the quality of a presented recipe without considering novelty, often preferring prior success to exploring new possibilities. The goal of our system is not only to produce a good recipe, but also to produce a *creative* one. This requires high quality as well as the development of *novel* artefacts.

2 PIERRE

Recipe generation is a complicated task that requires not only precise amounts of ingredients, but also explicit directions for preparing, combining, and cooking the ingredients. To focus on the foundational task of the type and amount of ingredients, we restrict our focus to recipes (specifically soups, stews, and chilis) that can be cooked in a crockpot. Crockpot recipes simplify the cooking process to essentially determining a set of ingredients to be cooked together.

We introduce a novel recipe generation system, PIERRE (Pseudo-Intelligent Evolutionary Real-time Recipe Engine), which, given access to existing recipes, learns to produce new crockpot recipes. PIERRE is composed primarily of two modules, for handling evaluation and generation, respectively. Each of these components takes input from an inspiring set and each is involved in producing recipes to send to the presentation module, as shown in Figure 1. In addition, the system interacts with the web, both acquiring knowledge from online databases and (potentially) publishing created recipes.

2.1 Inspiring Set

The inspiring set contains 4,748 soup, stew, and chili recipes gathered from popular online recipe websites[1]. From these recipes we manually created both a list of measurements and ingredients in order to parse recipes into a consistent format. This parsing enabled 1) grouping identical ingredients under a common name, 2) grouping similar ingredients at several levels, and 3) gathering statistics (including min, max, mean, variance, and frequency) about ingredients and ingredient groups across the inspiring set. Recipes in the inspiring set are normalized to 100 ounces.

The database of ingredients was explicitly partitioned into a hierarchy in which similar ingredients were grouped at a *sub*-level and these ingredient groups were further grouped at a *super*-level. For example, as shown in Figure 2, the super-group *Fruits and Vegetables* is composed of the subgroups *Beans*, *Fruits*, *Leafy Vegetables*, and others. The subgroup of *Beans* includes many different types of beans including *Butter Beans*, *Red Kidney Beans*, *Garbanzo Beans*, and others.

Statistics are kept for each ingredient, including minimum, maximum, average, and standard deviation for the amount of the ingredient, as well as the probability of the ingredient occurring in an inspiring set recipe. These statistics are also aggregated at the sub- and super-group level, enabling comparison and evaluation of recipes at different levels of abstraction. In addition, gathering statistics at the group level provides for smoothing amounts for rare ingredients. Each statistic ω (min, max, mean, standard deviation, or frequency) for ingredients occurring less than a threshold in the set is linearly interpolated with the corresponding statistic of the sub-group, according to the following:

$$\omega = \begin{cases} \left(\frac{\alpha}{\alpha+\beta}\right) x + \left(\frac{\beta}{\alpha+\beta}\right) \xi & \text{if } \alpha < \theta \\ x & \text{if } \alpha \geq \theta \end{cases}$$

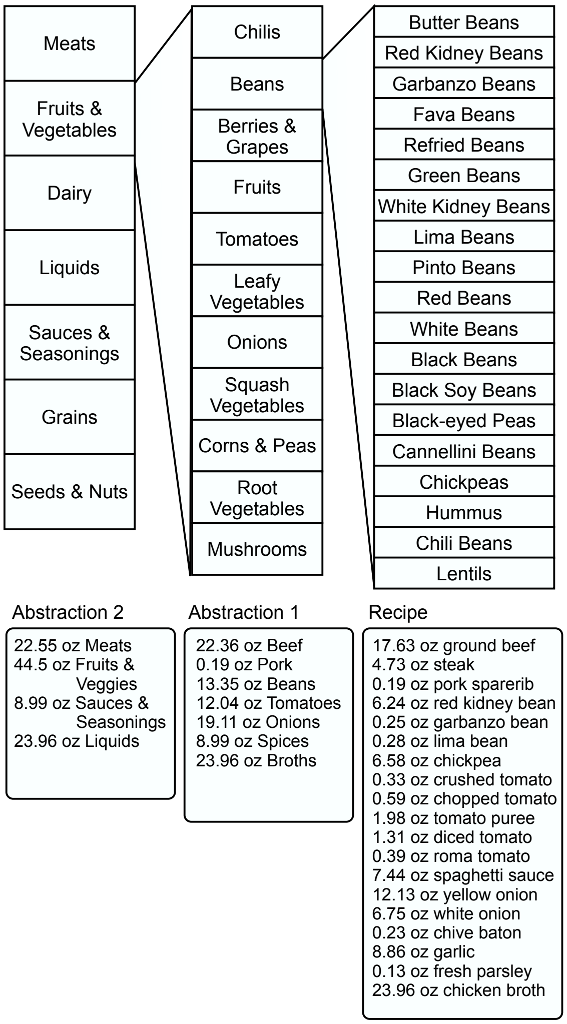
where x is the statistic of the ingredient, ξ is the statistic of the sub-group, α is the number of times the ingredient occurs in the inspiring set, β is the number of times any of the sub-group ingredients occur in the inspiring set, and the threshold θ is set to 100.

The inspiring set is used differently for generation than it is for evaluation. During artefact generation (Section 2.2) the inspiring set determines the initial population used for the genetic algorithm. During artefact evaluation (Section 2.3) the inspiring set determines which recipes and ratings are used as training examples for the multi-layer perceptron (MLP). Since the inspiring set is used in multiple ways, employing a different inspiring set for generating artefacts than the one used to evaluate artefacts can have useful effects.

2.2 Generation

PIERRE generates new recipes using a genetic algorithm acting on a population of recipes, each composed of a list of ingredients. The population is initialized by choosing recipes uniformly at random from the inspiring set, and the

[1] www.foodnetwork.com and www.allrecipes.com



22.55 oz Meats
44.5 oz Fruits & Veggies
8.99 oz Sauces & Seasonings
23.96 oz Liquids

22.36 oz Beef
0.19 oz Pork
13.35 oz Beans
12.04 oz Tomatoes
19.11 oz Onions
8.99 oz Spices
23.96 oz Broths

17.63 oz ground beef
4.73 oz steak
0.19 oz pork sparerib
6.24 oz red kidney bean
0.25 oz garbanzo bean
0.28 oz lima bean
6.58 oz chickpea
0.33 oz crushed tomato
0.59 oz chopped tomato
1.98 oz tomato puree
1.31 oz diced tomato
0.39 oz roma tomato
7.44 oz spaghetti sauce
12.13 oz yellow onion
6.75 oz white onion
0.23 oz chive baton
8.86 oz garlic
0.13 oz fresh parsley
23.96 oz chicken broth

Figure 2: Above, a view of the ingredient hierarchy, showing the super-group (left), sub-group (middle), and ingredient (right) levels of abstraction. The *Fruits & Vegetables* super-group is expanded to show its sub-groups, including *Beans*, which is expanded to show its ingredients. Below, an example recipe is shown as it would appear at each level of abstraction.

fitness of each recipe is evaluated using the MLP evaluator described in Section 2.3. To produce each generation, a number of new recipes are generated equal to the number of recipes in the population. For each new recipe, two recipes are selected, with probability proportional to their fitness, for genetic crossover. The crossover is performed by randomly selecting a pivot index in the ingredient list of each recipe, thus dividing each recipe into two sub-lists of ingredients. A new recipe is then created by combining the first sub-list of the first recipe with the second sub-list of the second recipe.

After crossover, each recipe is subject to some probability of mutation. If a mutation occurs, the type of mutation is selected uniformly from the following choices:

- *Change of ingredient amount.* An ingredient is selected uniformly at random from the recipe, and its quantity is set to a new value drawn from a normal distribution that is parameterized by the mean and standard deviation of that ingredient's amount as determined from the inspiring set.
- *Change of one ingredient to another.* An ingredient is selected uniformly at random from the recipe, and is changed to another ingredient from the same super-group, chosen uniformly at random. The amount of the ingredient does not change.
- *Addition of ingredient.* An ingredient is selected uniformly at random from the database and inserted into a random location (chosen uniformly) in the recipe's ingredient list. The amount of the new ingredient is determined by a draw from a normal distribution parameterized by the mean and standard deviation of the ingredient amount as determined from the inspiring set.
- *Deletion of ingredient.* An ingredient is selected uniformly at random and removed from the recipe.

At the completion of each iteration, evolved recipes are re-normalized to 100 ounces for equal comparison to other recipes. The next generation is then selected by taking the top 50% (highest fitness) of the previous generation and the top 50% of the newly generated recipes. The rest of the recipes are discarded, keeping the population size constant.

Recipes 1 and 2 were generated using this process and were among those prepared, cooked, and fed to others by the authors. To produce these recipes, a population size of 150 recipes was allowed to evolve for 50 generations with a mutation rate of 40%.

2.3 Evaluation

To assess the quality of recipes, PIERRE uses an interpolation of two MLPs. Taking advantage of the (online) public user ratings of the recipes in the inspiring set, these MLPs perform a regression of the user rating based on the amount of different ingredients. The two MLPs are trained at different levels of abstraction within our ingredient hierarchy, with one operating at the super-group level and the other at the sub-group level. Thus, the model at the higher level of abstraction attempts to learn the proper relationship of major groups (meats, liquid, spices, etc), and the other model works to model the correct amounts of divisions within those groups.

Because we assume any recipe from the online websites is of relatively good quality, regardless of its user rating, we supplemented the training set with randomly constructed recipes given a rating of 0. These negative examples enabled the learner to discriminate between invalid random recipes and the valid ones, created by actual people.

Each MLP has an input layer consisting of real-valued nodes that encode the amount (in ounces) of each super-group (sub-group), a hidden layer consisting of 16 hidden nodes and a single real-valued output node that encodes the rating (between 0 and 1). The MLP weights are trained (with a learning rate of 0.01) until there is no measurable improvement in accuracy on a held out validation data set (consisting

Recipe 1 Divine water with sirloin

Ingredients:
2.35 cups - water
2.07 cups - yellow onion
1.76 cups - black bean
1.43 cups - stewed tomato
10.71 ounces - steak
10.68 ounces - ground beef
0.72 cup - salsa
0.66 cup - chicken broth
3.01 tablespoons - emeril's southwest essence
0.87 ounce - veal
1.22 tablespoons - white onion
1.22 tablespoons - diced tomato
1.17 tablespoons - red kidney bean
2.79 teaspoons - sambal oelek
0.22 clove - garlic
2.28 teaspoons - white bean
1.83 teaspoons - corn oil
0.29 ounce - pancetta
1.67 teaspoons - mirin
1.51 dashes - tom yam hot and sour paste
1.46 dashes - worcestershire
0.12 ounce - bologna

Directions: Combine ingredients and bring to boil. Reduce heat and simmer until done, stirring occasionally. Serve piping hot and enjoy.

Recipe 2 Exotic beefy bean

Ingredients:
2.2 cups - pinto bean
1.09 pounds - ground beef
1.6 cups - white onion
1.16 cups - diced tomato
1.13 cups - water
1.11 cups - chicken broth
0.77 cup - vegetable broth
0.63 cup - chile sauce
2.74 ounces - pork sausage
4.51 tablespoons - salsa
3.39 tablespoons - stewed tomato
1.43 ounces - chicken thigh
2.5 tablespoons - olive oil
1.09 ounces - hen
0.34 whole - red bell pepper
1.25 tablespoons - lentil
1.16 tablespoons - chopped tomato
2.87 teaspoons - red onion
2.03 teaspoons - garbanzo bean
1.65 teaspoons - cannellini bean
0.26 slice - bacon

Directions: Combine ingredients and bring to boil. Reduce heat and simmer until done, stirring occasionally. Serve piping hot and enjoy.

of 20% of the recipes) for 50 epochs. The set of weights used for evaluating generated recipes are those that performed the best on the validation data set.

2.4 Presentation

Colton (2008) has suggested that *perception* plays a critical role in the attribution of creativity. In other words, a computationally creative system could (and possibly must) take some responsibility to engender a perception of creativity.

In an attempt to help facilitate such a perception of its artefacts, PIERRE contains a module for recipe presentation. First, the module formats the recipe for human readability. Ingredient quantities are stored internally in ounces, but when rendering recipes for presentation, the ingredients are sorted by amount and then formatted using more traditional measurements, such as cups, teaspoons, dashes, and drops. Recipes are presented in a familiar way, just as they might appear in a common cookbook.

Second, the presentation module generates a recipe name. Standard recipes always have a name of some sort. While this task could be a complete work by itself, we implemented a simple name generation routine that produces names in the following the format: *[prefix] [ingredients] [suffix]*. This simple generation scheme produces names such as “Homestyle broccoli over beef blend” or “Spicy chicken with carrots surprise.” The components of the name are based on prominent recipe ingredients and the presence of spicy or sweet ingredients. This simple approach creates names that range from reasonable to humorous.

3 EmPIERREical Results

To our knowledge, no other creative system has been designed to work in the recipe domain. As such, traditional concepts are highlighted in a new context. This new perspective admits additional analysis of the merits and nuances of theoretical ideas that have become generally accepted by the community. Here we evaluate the system with different combinations of inspiring sets, with and without a direct measure for typicality, and with and without the hierarchical definition of an artefact.

We measure novelty in a recipe by counting new combinations of (known) ingredients, n-grams. An n-gram is a combination of n ingredients. For example, a 2-gram would be *water-garlic*. A *rare* n-gram is an n-gram that does not occur in the inspiring set and does not contain a rare $(n-1)$-gram as a sub-combination (e.g., 4-grams containing rare 3-grams or, recursively, rare 2-grams are not included in the count of rare 4-grams). We define the *rare n-gram ratio* ρ_r^n for a specific recipe r as

$$\rho_r^n = \frac{\lambda_r^n}{\tau_r^n}$$

where τ_r^n is the total number of n-grams in r and λ_r^n is the number of those n-grams that are rare.

As another view of novelty, we consider a graph of ingredient amounts, which creates a visual profile of the type of recipes generated by the system. This comparison of visual profiles was inspired by Faria and de Oliveira's use of a similar method in measuring aesthetic distances between document templates and generated document artefacts (Faria and de Oliveira 2006), and we found that it was easy to compare the outputs of the system based on the profiles that it generated.

3.1 Different Inspiring Sets for Evaluation

As mentioned, PIERRE can have different inspiring sets for both artefact generation and artefact evaluation. Thus the artefact initially generated would be inspired by one set of artefacts, but fitness would be determined by a fitness function inspired by a different set of artefacts. Using a combination of inspiring sets in the generative process hints at an idea which Buchanan identifies as "transfer" or knowledge sharing (Buchanan 2001), which refers to the notion that where two problems have simple, heterogenous representations, greater creativity can be achieved by transferring knowledge from one problem area to another. Although developing recipes from different inspiring sets may not constitute different problems in the same way as intended by Buchanan, the concepts and methods used by humans to develop recipes in one inspiring set may differ greatly from the concepts and methods used to develop recipes in a different culinary genre. Thus the knowledge used in the composition of artefacts in one inspiring set is introduced in the generation of new artefacts in a different domain, resulting in potentially greater creativity.

We experimented with various combinations of two inspiring sets. The first inspiring set included 4,748 soup, stew, and chili recipes crawled from the web (referred to as the "full" inspiring set). The second set is a subset of the first, including only the 594 chili recipes. The chili recipes were longer on average than the full recipes (13.97 ingredients as compared to 11.88 ingredients). We found no significant results from varying the generator's inspiring set therefore all reported experiments were conducted with a generator trained with the full inspiring set. We found that the recipes produced using the chili inspiring set to train the evaluator (hereafter referred to as the "chili evaluator") had a higher ratio of rare 2-grams and 3-grams (see blue lines in Figure 3) than those produced using the full inspiring set to train the evaluator (hereafter referred to as the "full evaluator", see red lines in Figure 3), and a relatively lower ratio of rare 4-grams and 5-grams. Because the system is using different inspiring sets to generate and evaluate recipes, it alters the original recipes to look more like the recipes found in the evaluator's inspiring set. In this context, generic soups or stews are being modified to look more like chilis. The resulting chilis retain some of the characteristics of the generic soups and stews, resulting in more novel combinations of ingredients and flavors (for chilis).

Systems which trained the evaluator with chili recipes produced recipes with a "chili" profile, as evidenced by more meat and vegetables, and less dairy and liquids (see blue lines in Figure 4). Systems which trained the evaluator with full recipes produced recipes with a marked "full" profile (red lines). This discovery suggests that a system's creativity can be guided through the use of different inspiring sets. Combining the use of different inspiring sets could introduce different flavor profiles, and allow the system to explore new parts of the recipe space.

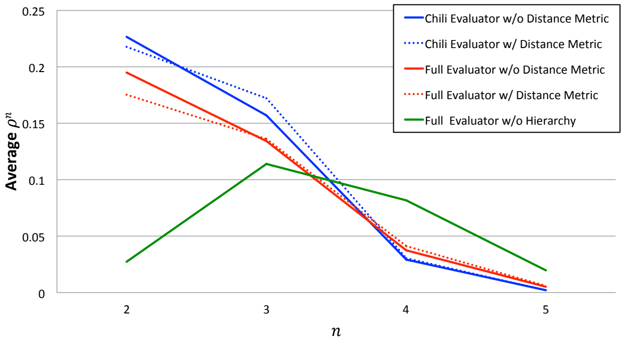


Figure 3: Average (over r) rare n-gram ratio for various values of n. Higher ratio values indicate increased novelty, with the chili evaluator producing the most novelty. Omitting the hierarchy noticeably reduces novelty, whereas including the distance metric has little effect.

3.2 Elimination of Explicit Typicality Metrics in the Fitness Function

We tested PIERRE with and without an explicit distance metric to essentially model a Wundt curve (Saunders and Gero 2001), promoting the generation of recipes that were neither too novel nor too typical. Although the theory can be interpreted to require an explicit evaluation of typicality (Ritchie 2007), in our experiments we found that removing the distance metric from our evaluation has no significant effect on the typicality or the novelty of our recipes (see the dotted lines in Figures 3 and 4). Explicitly measuring typicality is not necessary if typicality is implicitly modeled in the artefact generation process. In our system, ingredient quantities and ingredient counts were generated based on statistics found in the inspiring set. In addition, typicality is

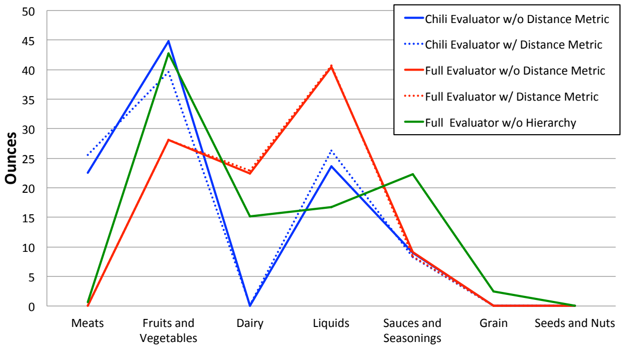


Figure 4: Ingredient amount (in ounces) of each supergroup. Different evaluators result in unique flavor profiles demonstrated by a visibly different recipe make-up. Omitting the hierarchy results in less extreme peaks and valleys and including the distance metric has little effect.

inherently imposed by both the generator and the evaluator. The generator selects new values based on normal distributions parameterized by statistics of the inspiring set, and if any recipe strays too far from what is typical, the evaluator assigns it a low fitness score.

3.3 Explicit modeling of the Ingredient Hierarchy

While modeling artefacts hierarchically perhaps seems like an obvious improvement for many creative systems, we compare our system with and without the hierarchy to validate that intuition. In terms of novelty (Figure 3), the recipes produced without the explicit hierarchical model (green line) exhibit fewer rare n-grams than the recipes produced using the hierarchy. Thus, the system is more capable of generating novel recipe combinations with the hierarchical model. This added novelty comes from the extra information that is introduced through the hierarchy. For example, when the system is generating a recipe, it can now know that, rather than a specific liquid, it really needs a type of liquid, thus allowing an interplay between typicality (maintaining the same sub/super-group) and novelty (trying a new member of the group). The system can then search for creative alternatives for the generic liquid that it needs to include in the recipe.

In addition, note that the ingredient amount profile of recipes created without the explicit hierarchy model (see the red and blue lines in Figure 4) exhibits less pronounced peaks or troughs than profiles for recipes generated using the hierarchy, suggesting that the hierarchy informs the system in generating recipes more characteristic or typical of a specific recipe genre. These results suggest that interpolated assessment of creative artefacts at multiple levels of abstraction is more effective at facilitating creativity than a unilateral assessment.

4 Discussion

One of the major contributions of this work is to provide a(nother) concrete implementation of many of the theoretical components called for in the literature, focusing on the area of computational recipe generation.

In doing so, we have presented a number of useful concepts, including the use of different inspiring sets for generation and evaluation, the implicit modeling of typicality (versus the notion of explicitly measuring it), and the abstraction of artefacts into a hierarchy for explicit use in both evaluation and generation.

The idea of evaluation-specific inspiring sets suggests a natural step towards meta-level creativity by providing a mechanism for changing evaluation criteria. In the context of recipe generation, this could include the ability for the system to change its "taste" over time, or, to use different inspiring sets to create different flavor profiles. Thus, just as the system has a more pronounced chili profile for its recipes when using the chili inspiring set for the evaluator, it could induce other types of profiles using other types of inspiring sets for evaluation and this affinity could vary over time. In general, the culinary arts provide a rich framework for varying preferences such as spice, sweetness, and texture, that could all be considered at the meta-level.

Table 1: Presentation survey results. Formatting recipes resulted in no significant difference.

Question	No Format	Format
Assuming I cooked on a regular basis, I would cook this recipe.	2.66	2.27
I think this recipe is creative.	3.59	3.28
I think this recipe is novel.	3.21	3.12
I think this recipe would be difficult to invent.	3.15	3.46
This recipe looks like it would taste different than anything I've previously tasted.	3.47	3.52
This recipe looks like it would taste good.	3.43	3.14

One significant area for future work is to incorporate the notion of goals and plans. As is, the system has a single goal: to create a high quality, high novelty chili. The different portions of the recipe space being explored by different parts of the population from the genetic algorithm could be seen as exploring different versions of that goal (for example, one part of the population would be predominantly chicken chili while another part would be predominantly vegetarian chili). However, the system would have more creative power if it could create and refine its own goal as it explores. Given user input, or even other factors (such as weather), the system could change its goal over time to be more appropriate to the given context.

In an attempt to assess the effectiveness of the presentation module, we hypothesized that the (admittedly simple) presentation effected by the system would make the artefacts more pleasing (than the raw, system versions of the recipes) to humans, and thus would increase the perceived creativity of the system. Thirty-eight participants were randomly given one of two surveys and asked to rate each of five recipes according to certain criteria. The first survey contained five unformatted recipes (no title, ingredients measured in ounces, without rounded quantities), whereas the second survey contained the formatted versions of the same five recipes (title, standard measurements, and rounded quantities). Contrary to our hypothesis, no statistically significant difference existed between the responses in the two groups (see Table 1).

Analysis of the survey lead us to an interesting (though perhaps retrospectively obvious) realization that the survey had not explicitly asked/forced the respondents to consider the creativity of the *written* recipe. Although the written recipes were quite different in each of the two cases, the *cooked* form of the recipes was the same for both formats. The effects of presentation on perceived creativity depend on which form of the artefact is being evaluated. In other words, were the survey participants evaluating the quality of the written recipe, or were they considering more what the cooked version would taste like? As another example of this idea, consider a system which generates musical scores. Is the creativity of artefacts produced by such a system determined from the written score or from the music that is heard when a musician plays the score?

Though this question may seem trivial, consider that when 10 of PIERRE's recipes were posted on Food.com, the online community was outraged enough by some of the ingredient quantities (e.g., a dash of green beans)—which, though absurd by human standards, would not negatively affect taste—that even without considering the quality, or taste, of the cooked recipes, they removed the recipes from the site and suspended our account. The lack of typicality in the *written* recipe was condemned without considering that if cooked, the resulting chili would be considered typical (and possibly even tasty) by any reasonable measure.

We assert that many creative domains admit both a "written" and a "cooked" version of the artefact. Recognizing the distinction between the two may be essential to eliciting quantitative (or even qualitative) standards for evaluating creativity. Much of the work in computational creativity to date appears to have been on one level or the other—on the level of "written" artefacts perhaps because it is difficult to work at all at the "cooked" level or, possibly, because it is not obvious that there are, in fact, two distinct levels of artefact representation (e.g. visual art, perhaps?). Music is another good example of this phenomena. The sheet music is a written artefact, with instructions on how to produce the "cooked" artefact (or the actual melody). If there is no way to listen to the melody being played, then the artefact must be evaluated at the "written" level (using the sheet music alone). The idea of building modules which simulate forms of human perception has been explored to some extent in fields like computer vision, (audio) signal processing, haptic systems and the like, but in the context of computational creativity this sort of dual representation and evaluation is still largely unexplored at present, and this could represent a significant hurdle to establishing conventional tactics in evaluating computational creativity research (Jordanous 2011).

References

Binsted, K., and Ritchie, G. 1997. Computational rules for generating punning riddles. *HUMOR-International Journal of Humor Research* 10(1):25–76.

Buchanan, B. 2001. Creativity at the metalevel: AAAI-2000 presidential address. *AI Magazine* 22(3):13.

Cohen, H. 1999. Colouring without seeing: a problem in machine creativity. *AISB Quarterly* 102:26–35.

Colton, S. 2002. *Automated Theory Formation in Pure Mathematics*. Springer.

Colton, S. 2008. Creativity versus the perception of creativity in computational systems. In *Proceedings of the AAAI Spring Symposium on Creative Systems*.

Faria, A., and de Oliveira, J. 2006. Measuring aesthetic distance between document templates and instances. In *Proceedings of the ACM Symposium on Document Engineering*, 13–21. ACM.

Gervás, P. 2000. Wasp: Evaluation of different strategies for the automatic generation of spanish verse. In *Proceedings of the AISB-00 Symposium on Creative & Cultural Aspects of AI*, 93–100.

Gervás, P. 2001. An expert system for the composition of formal spanish poetry. *Knowledge-Based Systems* 14(3):181–188.

Gervás, P. 2011. Dynamic inspiring sets for sustained novelty in poetry generation. In *Proceedings of the Second International Conference on Computational Creativity*, 111–116.

Hammond, K. 1986. Chef: A model of case-based planning. In *Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI-86)*, volume 1, 267–271.

Hinrichs, T. 1992. *Problem solving in open worlds: A case study in design*. Lawrence Erlbaum Associates.

Jordanous, A. 2010. A fitness function for creativity in jazz improvisation and beyond. In *Proceedings of the First International Conference on Computational Creativity*, 223–227.

Jordanous, A. 2011. Evaluating evaluation: Assessing progress in computational creativity research. In *Proceedings of the Second International Conference on Computational Creativity*, 102–107.

Lewis, G. 2000. Too many notes: Computers, complexity and culture in voyager. *Leonardo Music Journal* 10:33–39.

Monteith, K.; Francisco, V.; Martinez, T.; Gervás, P.; and Ventura, D. 2011. Automatic generation of emotionally-targeted soundtracks. In *Proceedings of the 2nd International Conference in Computational Creativity*, 60–62.

Norton, D.; Heath, D.; and Ventura, D. 2011. Autonomously creating quality images. In *Proceedings of the 2nd International Conference in Computational Creativity*, 10–15.

Pérez y Pérez, R., and Sharples, M. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence* 13(2):119–139.

Pérez y Pérez, R. 2007. Employing emotions to drive plot generation in a computer-based storyteller. *Cognitive Systems Research* 8(2):89–109.

Ritchie, G., and Hanna, F. 1984. AM: A case study in AI methodology. *Artificial Intelligence* 23(3):249–268.

Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17(1):67–99.

Saunders, R., and Gero, J. 2001. The digital clockwork muse: A computational model of aesthetic evolution. In *Proceedings of the AISB*, volume 1, 12–21.

Stock, O., and Strapparava, C. 2005. The act of creating humorous acronyms. *Applied Artificial Intelligence* 19(2):137–151.

Veale, T., and Hao, Y. 2007. Comprehending and generating apt metaphors: a web-driven, case-based approach to figurative language. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, volume 2, 1471–1476. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Wiggins, G. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19(7):449–458.