CO659 Project: Gourmake
Project Report

Chris Bailey

`cb661@kent.ac.uk`

School of Computing
University of Kent
United Kingdom

May 6, 2018

Abstract

This paper describes the development, operation, conclusion and my reflection on a computational creativity system which invents recipes, henceforth referred to as 'Gourmake'.

At the time of writing, the Gourmake system is arguably successful in achieving its goal of being able to produce human-like recipes drawing only from the encoding of ingredient information and a learning set, much like the PIERRE system described later.

Gourmake is also an attempt at declaratively generating creative artefacts, and despite several failings in the implementation due to misjudgments and time constraints, Gourmake is currently able to generate what appear to be rather novel, inventive and legitimate recipes.

1 Introduction

The primary goal of Gourmake was originally to be able to create simple and human-like recipes drawing on no knowledge other than basic information about ingredients and methods of preperation — whether or not Gourmake has achieved this is up to interpretation but at the time of writing Gourmake can and does generally produce human-like recipes.

Gourmake does this by drawing inspiration from existing recipes, which are heavily templatised at varying levels of abstraction, allowing essentially for Gourmake to autonomously fill in the gaps by cross-referencing categorisations of different ingredients and finding suitable ones to use; which whilst simple, is very extensible and produces surprisingly original and 'thoughtful' results.

This paper goes on to describe the background research which challenged original ideas about how Gourmake ought to be designed, as well as key implementation details which help Gourmake successfully achieve its primary goals. This paper also discusses the limitations of Gourmake, how to improve Gourmake and the future of Gourmake.

2 Background

Gourmake functions in a way inspired by the PIERRE system[0], creating recipes in a layered abstracted approach and as a result can create novel, interesting but more importantly also rather edible artefacts.

At a high level, Gourmake, like the PIERRE system[0] creates dises from an inspiring set of recipes, as well as many layers of abstraction for ingredients which are to be used in these recipes. We will explore these abstractions and ideas underlying Gourmake's generative process futher in the paper.

3 Methodoly and Design

3.1 Implementation and design of Gourmake

Gourmake was designed as a multi-agent system consisting of several autonomous agents, all of which play certain creative roles which come together to produce Gourmake's artefacts. This design was chosen primarily to abstract away different procedures in recipe construction and to be easily extensible. More agents which provide certain functionality can easily be added and plugged into the creative process if needed.

Gourmake is at a high level, composed primarily of two agents, one agent specialises in parsing our recipe database and finding correlations and categorisations between ingredients allowing us to query for any number of random ingredients given arbitary constraints. The other agent specialises in understanding the recipe database fed into Gourmake, and essentially coordinates different agents together to be able to produce output.

Because Gourmake deals in multiple agents which should be seperated from eachother, Gourmake is written in Erlang and utilises Erlang's underlying Actor Model message passing mechanisms to communicate between individual agents. This also allows us several benefits such as easy scalability and concurrency support making Gourmake ideal for a web-app or web-frontend allowing people to easily request new recipes on the fly. On a live system, Gourmake's agents run indefinitely in the background, but Erlang's REPL allows easily debuggability and testing of the system too which is another reason why this technology was chosen.

3.1.1 Ingredient Database Agent

The Ingredient Database Agent is, as stated, responsible for handling, choosing and understanding individual recipes in Gourmake's creative process.

Gourmake is, at the time of writing, pre-loaded with knowledge of 629 ingredients (datamined from the BBC[1]) which is contained in a plaintext file read in by Gourmake. The file is essentially an Erlang term which is evaluated at runtime and works essentially the same as JSON. This datastructure maps a given ingredient name to arbitarily many different categories and cuisines associated with said ingredient which the Ingredient Database Agent processes to learn.

The categories and cuisines which are associated with individual Ingredients are represented by Erlang atoms and thus are only data labels which means adding new cuisines or categories to a given ingredient is as simple as modifying the file read in by Gourmake, nothing else needs to be done.

Because of this, any category or cuisine can be added allowing Gourmake the implicit ability to abstract the categorisation of its ingredient data to an arbitary level. Ingredients can be categorised however a given individual wants, be it as simple as *"A root vegetable"* to *"Something spicy"*.

Because boolean logic can be applied to categories and cuisines, unlike the PIERRE system, I've found it unneccessary to nest categories because a similar effect can be constructed to their examples by simply finding the union or intersection of the results of multiple categories.

Once this file has been parsed, the Ingredient Database Agent processes it to build an 'network' representation of all of the ingredients. This network subdivides individual ingredients into lookup tables for categories, cuisines and ingredient names which gives the Ingredient Database Agent the ability to quickly and efficiently search through our list of processed ingredients cusinewise, categorywise or ingredient namewise.

When other agents communicate with the Ingredient Database Agent, they can then provide constraints such as *"An aromatic spice, which isn't sweet nor a dessert, typically used in indian cuisine"*. After a series of filters and searches through our network, the Ingredient Database Agent returns one ingredient which best matches the given constraints.

If a constraint cannot be satisfied, the Ingredient Database Agent expands its search for similar categories, cuisines or simply weakens the constraints placed upon it until it successfully finds an ingredient. There is also a stochastic probability that the Ingredient database agent will ignore constraints or invent new ones, which results in less predicatble output.

There was more work to be done on the Ingredient Database agent, but this is discussed futher down.

3.1.2 Recipe Database Agent

The Recipe Database Agent is at a high level not dissimilar to the Ingredient Database Agent, except it does processing on a file containing an Erlang Term representing our 'Learning Set'.

This 'Learning Set' is a set of templatised recipes which encode human knowledge ready to be filled in. Currently, at the time of writing, the 'Learning Set' consists only of 16 examples. This was simply due to the fact that all of the datasets used in Gourmake are hand written to conform to the exact scheme created for Gourmake. However, even with only 16 example recipes in the 'Learning Set', the output of Gourmake always seems rather original and there is seldom much similarity between generated recipes.

The formatting of the 'Learning Set' is again, JSON-like, and a particular recipe encodes information about how it should be named, what abstract categories of ingredients should be used (as well as minimum and maximum quantities of each category, and contraints imposed upon a pending Ingredient-Database query) and valid cuisines a recipe can be tweaked to satisfy.

The Recipe Database Agent simply parses through the encoded template recipes, and makes decisions such as how many ingredients of a given category need to be selected and used for this recipe. That means that recipes vary not only in what ingredients are used but how many of each ingredient category are present. Gourmake does not yet provide any measurements for ingredients as there is no normalised weight system implemented in the Recipe or Ingredient databases yet but this was pending being added.

The Recipe Database Agent ultimately is the component which does all of the work; it communicates with agents such as the Naming Agent (which is responsible for naming recipes based on simply looking at an underlying recipe and selected ingredients), and the Ingredient Database Agent, coordinating them to produce an artefact.

3.1.3 Miscellaneous Components

Gourmake also utilises some miscellaneous components to improve the running of the system. All of our server agents are supervised by a higher level supervisor which can react to individual servers crashing. The supervisor automatically restarts such processes and ensures everything is running well.

There exists also a Naming Agent which names recipes based on the template provided by the recipe as well as the generated ingredient list produced by the Recipe Database Agent. For more personality and character, the Naming Agent also randomly selects a noun or adjective to prepend onto the generated recipe name allowing for names such as *"Ritchie's European inspired One-pan Farfalle with Beef Ribs, Radish and Sweetcorn"* or *"Knuth's Simple Emmental and Gruyere omellete"*.

3.2 Remaining work to be done

Despite the fact that Gourmake can successfully produce human-like output, and create at times rather sensible, original and inventive recipes, due to time constraints, much work which was intended to be complete isn't at the time of writing complete. The following features were planned to be implemented and will be implemented at a future time to improve Gourmake's generated artefacts.

3.2.1 Abstraction Layer for Ingredient Preparation

At the time of writing, Gourmake can abstract details about cuisine and categories of specific ingredients to allow it to select random yet 'thoughtful' ingredients for recipes, however, each step of a recipe is currently hard coded in the recipe templates which exist in our 'Learning Set'. This means that the individual steps for each recipe are fixed and cannot change.

For more varied output, Gourmake would need to be able to abstract how how to prepare certain ingredients and stochastically choose preparation methods based on cuisine and ingredients used rather than hard coding. This isn't a particular difficult task to implement, it just has not been implemented yet due to time constraints. Much like the Ingredient Database Agent, a Preparation Database Agent would need to be added which would encode information about how particular ingredients are best prepared, commonly prepared and have these also substituted into a recipe template.

3.2.2 Self-Evaluation of Produced Artefacts

Gourmake was originally going to use a method of generation based on Evolutionary Algorithms where the produced output of Gourmake would only be presented to a user after many generations of 'evolving'. The main issue with self-evaluation is finding a means of self-evaluating.

A proposed fitness function was to compare the similarity of chosen ingredients to those in other recipes, or to have real world example recipes from which recipes in the learning set are based off. The issue with this and quite literally the only reason this hasn't been done is because the learning set is currently so small.

Once this is implemented, we can gradually evolve more realistic looking recipes over time which would only increase the quality of Gourmake's output.

3.2.3 Improvements to Ingredient Database Agent

Utilising Self-Evaluation and having access to real-world recipes to learn from, a proposed change to the Ingredient Database agent is an improved probability to select ingredients which are used alongside already selected ingredients. Currently this isn't done and instead is implemented such that there is a low probability of substituting any ingredient with any ingredient in an adjacent category to the proposed ingredient.

4 Results

Despite the fact that many planned features were not implemented due to time constraints, Gourmake still arguably produces human-like artefacts which at times can appear to be quite creative and novel. Some raw examples of Gourmake's output are below *(more examples can be found on Gourmake's GitHub repo at **https://github.com/vereis/gourmake**)*:

```
# Gosling's Asian inspired Cod and New Potato Casserole with Radicchio and
  Creamed Coconut and Tartare Sauce
## Ingredients:
- New Potato
- Cod
- Radicchio
- Creamed Coconut
- Tartare Sauce
## Instructions:
1) Preheat oven to around 300 - 350 degrees.
2) Combine Cod and Radicchio in a mixing bowl and mix until even.
3) Then, add your Creamed Coconut and incorporate it into the contents of your
   mixing bowl
4) Move your mixed ingredients into a pot and cook on the stove until cooked
   through.
5) Finally, move your mixed ingredients into a baking tray and top it with your
   Creamed Coconut and Tartare Sauce.
6) Bake the contents of your baking tray for 20-30 minutes or until golden
   brown and enjoy.
```

You can clearly see that the generated name for the dish successfully sounds human-like and not neccessarily generated outside of the clumsy grammar when it comes to conjoining multiple items. In this particular case, the ingredients chosen successfully reflect what seems to be a valid and legitimate recipe.

Everything except the instructional text, which simply acts as glue, is stochastically generated and is the result of nothing but probability manipulation as defined by the Ingredient Database Agent's ingredient network as well as the Recipe Database Agent's probabalistically selected cuisines, ingredient categories and other constraints.

5 Evaluation

6 Conclusion