Felix Santiago

A Wine and Food Pairing Recommendation System

Introduction:

Everyone has been to a fancy restaurant at one point in their life or another. After ordering one's food, it is always intimidating to scan the page-long list of wines and their respective years, trying to decide which wine would best suit the meal. Chances are that most people would not call themselves expert wine connoisseurs. Thus, a knowledge based system can be utilized to aid the average person in determining proper food and wine combinations.

The system discussed in this paper outputs appropriate wine recommendations based upon user inputs regarding the type of meal and the ethnicity of the recipe. The system's knowledge includes seven types of wine: Cabernet, Chardonnay, Merlot, Pinot Noir, Riesling, Sauvignon Blanc, and Zinfandel. Using these seven wines, the system can make appropriate wine recommendations for foods from the following ethnicities: American, Caribbean, Chinese, French, Italian, Mexican, and Middle Eastern.

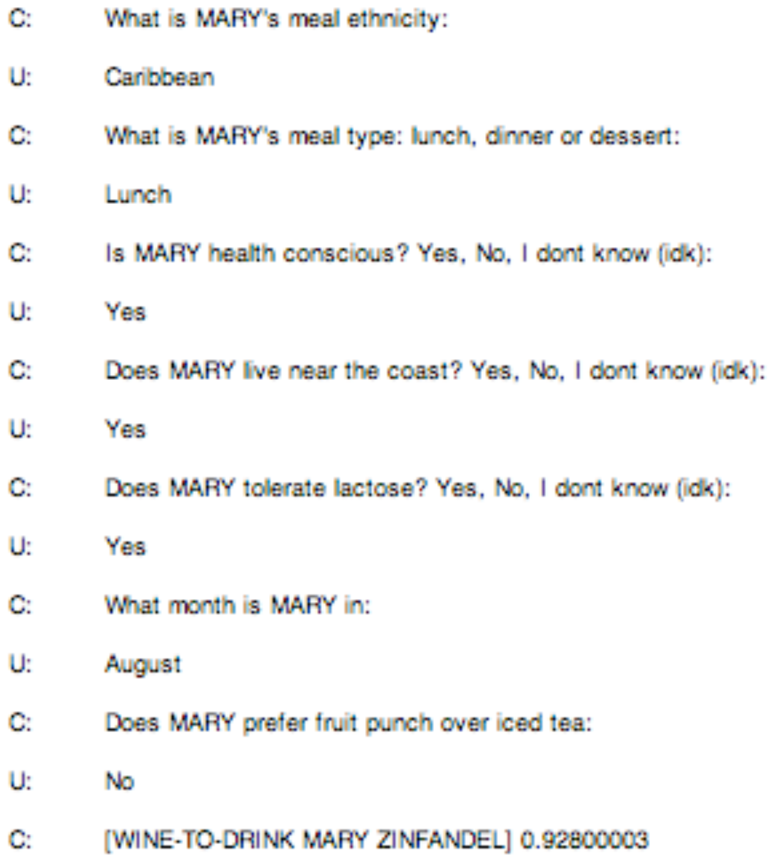
The system's task and knowledge base will be illustrated in further depth throughout the paper. Additional issues that will be addressed in this paper include an explicit example of the system's input and output, a discussion of the system's limitations, and an analysis of the successes and failures that were encountered during the project. The ending commentary will state what lessons were learned from this assignment.

System Task:

Our system's task is to recommend its user a wine that pairs up appropriately to the meal that will be consumed. In order to make a fitting wine recommendation, the user must input the answer to the questions that the system displays on the screen. Generally, the questions are as follows:

- What is the ethnicity of the restaurant?
- What is the meal type? (lunch, dinner, dessert)
- What month is it? (in order to infer the season and foods typical of that season)
- Does the user have any dietary concerns? (vegetarian, lactose intolerant, health conscious, etc)
- What type of flavors does the user prefer? (spicy, sweet, etc)

The questions that the system asks will not be the same in all cases. The questions may vary depending on the ethnicity of the food and meal type that the user inputs. It is important to note that our system does not simply match ingredients and preferences to a wine. The user is not asked to input the type of meat, sauce, or spices in the meal. Instead, the system utilizes the answers from these and other questions to infer the meal's ingredients and make a wine recommendation. Figure 1 provides an example of the system's general input interface.



```
C:      What is MARY's meal ethnicity:

U:      Caribbean

C:      What is MARY's meal type: lunch, dinner or dessert:

U:      Lunch

C:      Is MARY health conscious? Yes, No, I dont know (idk):

U:      Yes

C:      Does MARY live near the coast? Yes, No, I dont know (idk):

U:      Yes

C:      Does MARY tolerate lactose? Yes, No, I dont know (idk):

U:      Yes

C:      What month is MARY in:

U:      August

C:      Does MARY prefer fruit punch over iced tea:

U:      No

C:      [WINE-TO-DRINK MARY ZINFANDEL] 0.92800003
```

Figure 1: An example of the input interface

<u>Knowledge Base/Problem Solving Paradigm Overview:</u>

Our system utilizes JOSHUA, a rule based inference engine that we previously used during problem set 2. The system's task lends itself to a rule based system due to the ease in which a rule-based system infers knowledge (such as a wine recommendation) from given facts (such as the meat type and ingredients of a meal). Additionally the ease to transcribe knowledge and the familiarity with the software tools made us choose this particular knowledge representation

Originally the application was supposed to utilize a frame-based system along with rules as its knowledge representation system. After the rule portion of the application infers the ingredients of the meal, based on the user input, the characteristics of the meal are matched to preexisting frames that will then output a wine recommendation. Each wine would have its own frame or set of frames with different slot values representing features unique to that wine. We believed frames were ideal for our intended task, for they are faster and more efficient in the matching process than utilizing rules for the same purpose.

Based on the ethnicity inputted by the user, there are four main categories of rules involved in the system's operation:

- Rules inferring the type of meat
- Rules inferring the type of sauce
- Rules inferring the type of herbs and spices
- Rules inferring user preferences

Example problem walkthrough:

In order to infer the type of meat, sauce, herbs, and spices based on a certain ethnicity, certain generalizations had to be made concerning the typical dishes that each ethnicity produces. For instance, if the user inputs Italian as the ethnicity, the rules will indicate beef or poultry/seafood as the meat, since Italian dishes primarily include ground beef or seafood. In contrast, if the user inputs Indian as the ethnicity, the rules will infer that the meat is not beef because it is against Hindu religion to consume beef. Following the inference on the type of meat used in the meal, the system prompts the user to answer more questions in order to narrow down the possible wine choices.

The next question the program asks the user is the type of meal they will be eating. While lunch and dinner provide similar wine choices, an input declaring that the meal is dessert will automatically result in Riesling as the wine recommendation. Riesling will be recommended no matter what ethnicity is inputted because it is currently the only dessert wine in the system.

After determining the meat type, the system asks the user questions about their preferences and dietary restrictions in order to further narrow down the wine selections. For example, if the user entered French as the ethnicity and stated that the meal type was either lunch or dinner, the user would then be asked questions about the user's preferences. First it will ask whether or not the user is health conscious. If the user answers yes, the rules will assume that they chose the healthier chicken/seafood option over the pork/game bird option. If the user states that he is not health conscious, the system will assume he went with the pork/game bird option.

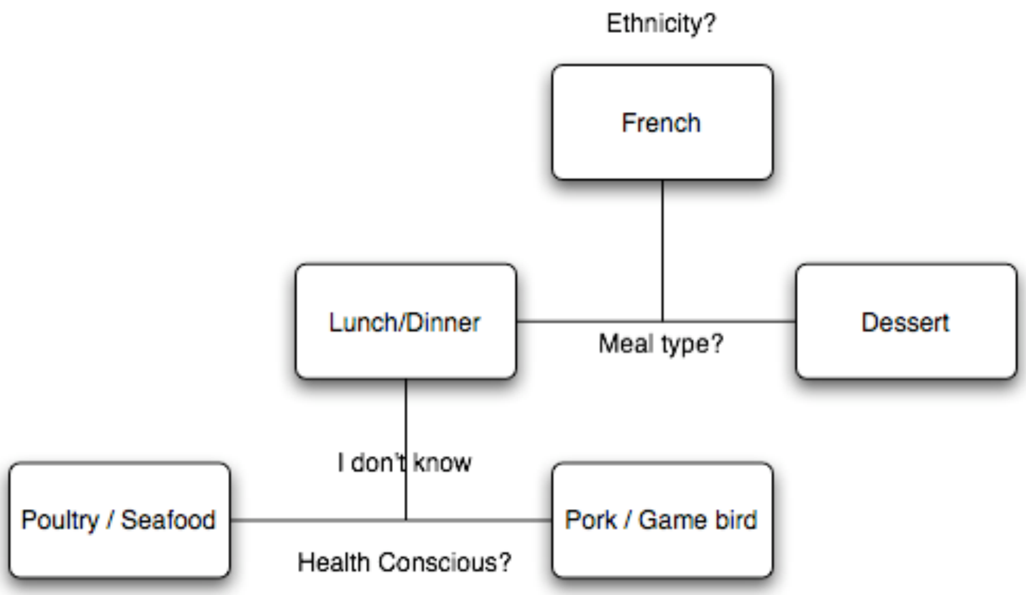


Figure 2: First set of questions for French ethnicity

If the system has inferred chicken/seafood as the meat type, it will ask the user whether or not he lives near or is visiting a restaurant near the coast. If the user says the restaurant is by the coast, the system will infer that the user ordered seafood. If the user says no, it is not by the coast, the system will infer that the user ordered chicken. Although this is a broad generalization, the rule is based on the fact that seafood is fresher by the coast and therefore seafood is the restaurant's specialty if it is located on the coast.

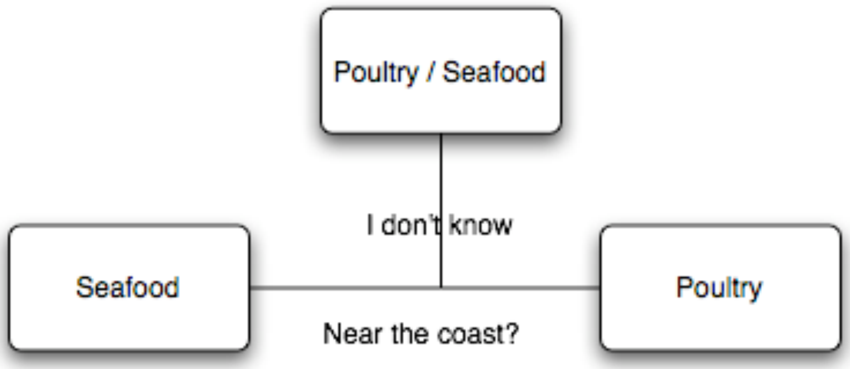


Figure 3: Costal question

In contrast, if the user declared that they were not health conscious and the system inferred the pork/game bird option, a different question would be asked. In this case, the user would then be asked if they enjoyed exotic tastes. If the user answered yes, the system would infer the game bird option. If the user answered no, the system would infer that they chose pork.

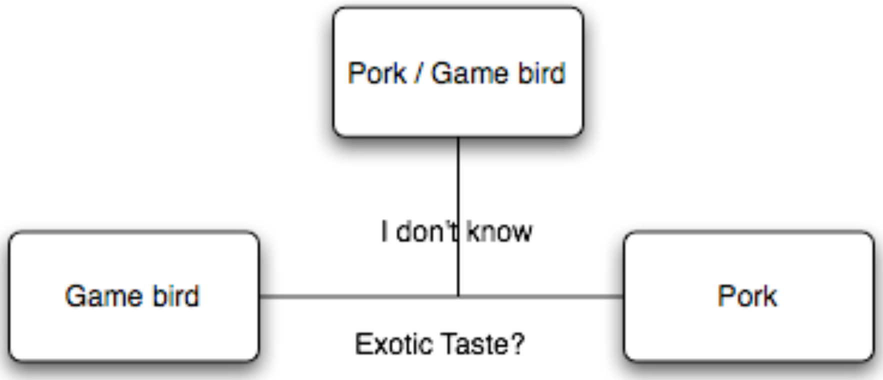


Figure 4: Exotic Taste question

Regardless of the meat type inferred, the user would next be asked if they lactose intolerant or not. If the user can tolerate lactose then the system would assume that the type of sauce might be cream-based. However, if the user is lactose intolerant then the system would then ask what month it is in order to determine what season it is and if tomatoes are in season so that it can be used as the base for the sauce. These two options in reality should be mutually exclusive since one cannot have both a cream-based sauce and a tomato-based sauce in the same meal, but, again, since we are generalizing these factors then allow both of these to co-exist in order to obtain more wine recommendations.

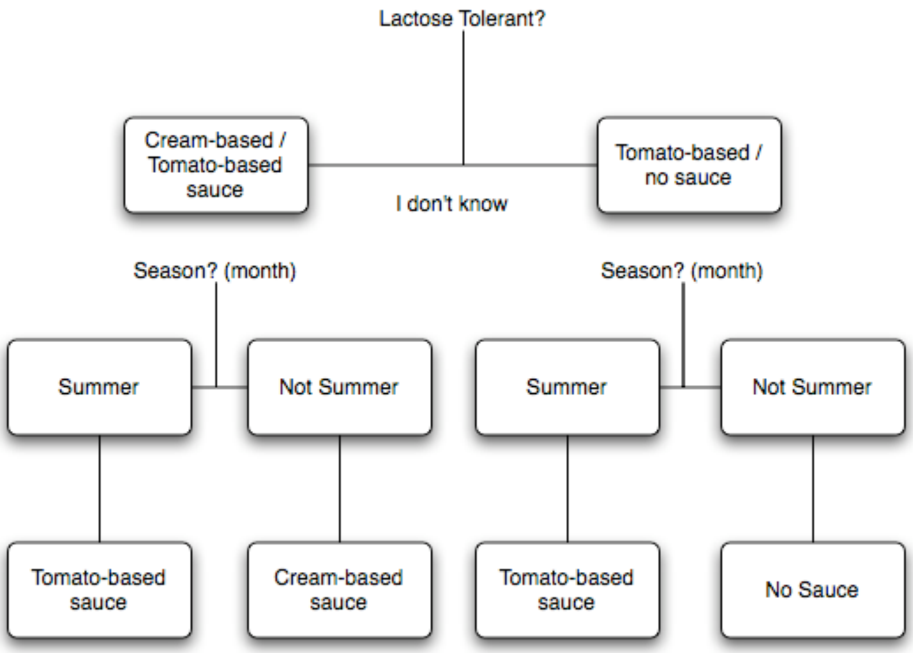


Figure 5: Questions relating sauce

After this, the user would then be asked if they were on a date. If the user answered yes, the system would assume they chose a dish without garlic. If the user answered no, the system would infer a dish with garlic. Although this is another broad generalization, this rule was written on the assumption that garlic makes your breath smell bad, which puts off dates and reduces the chances of a goodnight kiss.

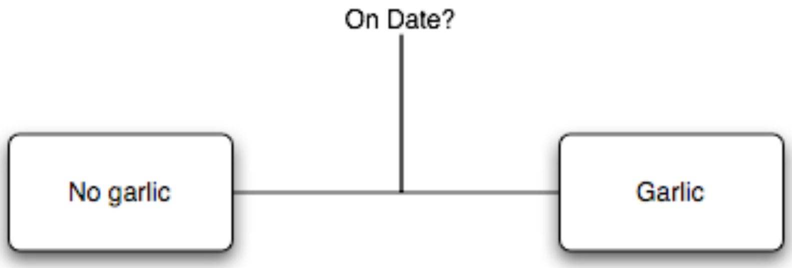


Figure 6: Date question

Next, the user is prompted to answer whether they would prefer to drink fruit punch or unsweetened iced tea. If the user specified fruit punch, the system would recommend a sweet wine if the previously made inferences haven't eliminated all the sweet wine options based on their frames. If the user chose unsweetened iced tea, the system would recommend a dry wine, if any dry wine options were still available.

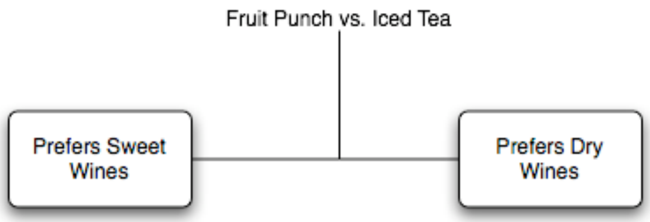


Figure 7: Drink preference question

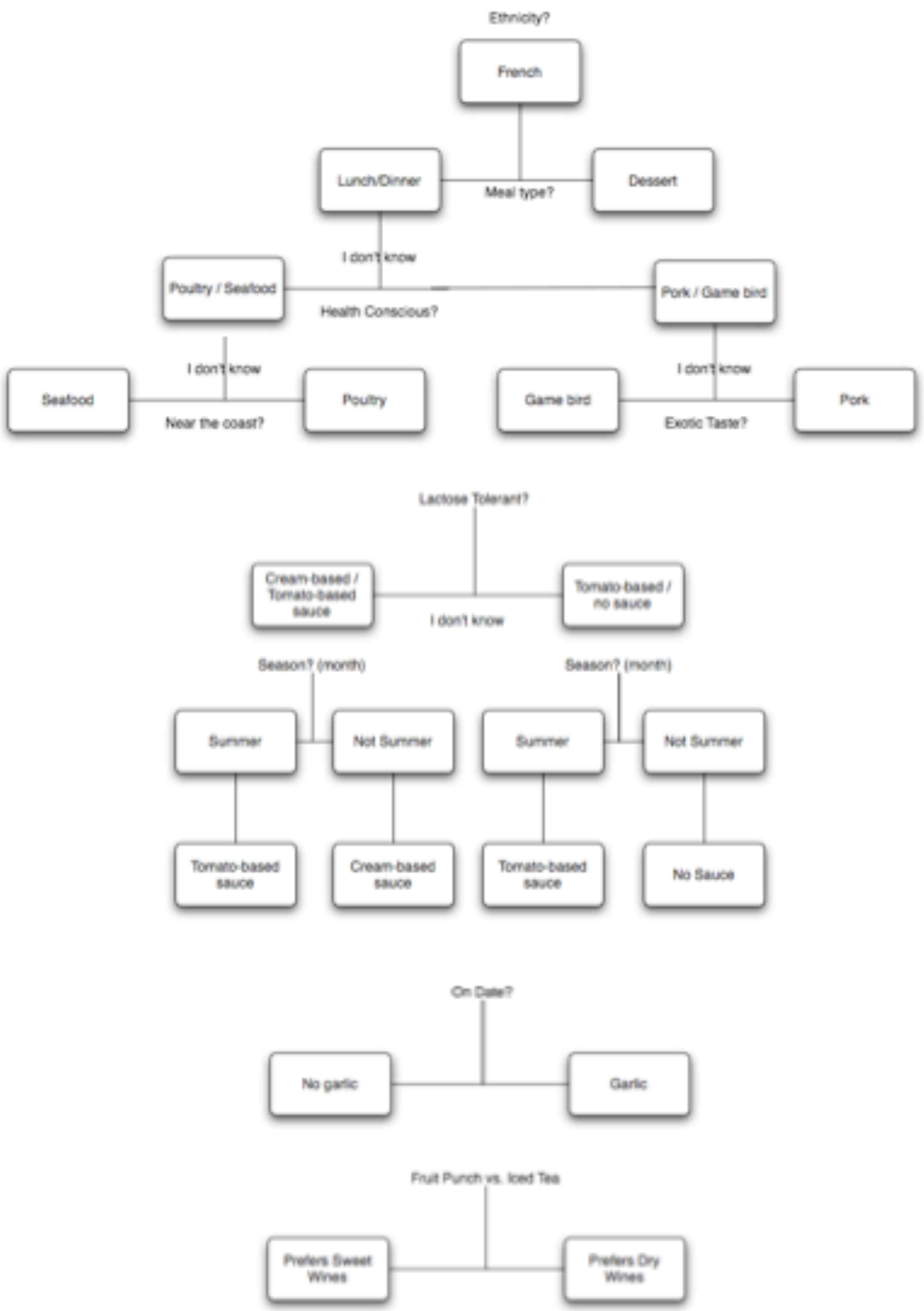


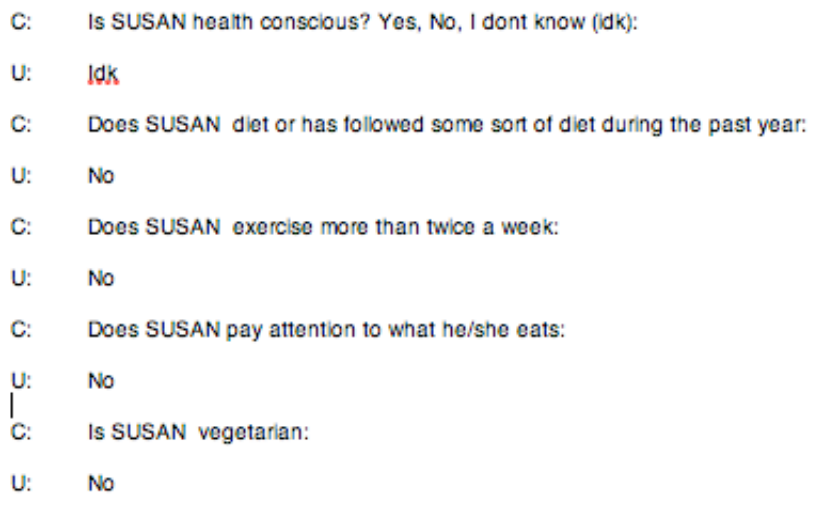
Figure 8: Complete set of questions with respective answers

As a specific example, if the user specified a Caribbean ethnicity for lunch, was health conscious, lived on the coast, tolerates lactose, is in August, and preferred iced tea, the system would recommend a Zinfandel with a 0.928 certainty based on those characteristics.

After declaring the ethnicity and meal type, the user is free to answer “I don’t know” to any of the preference questions. In the event that the user answers “I don’t know,” the system asks a series of questions that attempt to answer the preference question for the user. For example, if the system asks the user, “Do you like exotic tastes?” and the user does not know, the program will ask the following questions:

- Do you like to try new dishes?
- Do you like to try dishes with weird names?
- Do you like to travel?
- Do you like to hunt?

If the user responds “no” to each of these questions, the system comes to the conclusion that the user does not like exotic tastes and moves on to the next question based on that inference.



```
C:    Is SUSAN health conscious? Yes, No, I dont know (idk):
U:    Idk
C:    Does SUSAN  diet or has followed some sort of diet during the past year:
U:    No
C:    Does SUSAN  exercise more than twice a week:
U:    No
C:    Does SUSAN pay attention to what he/she eats:
U:    No
C:    Is SUSAN  vegetarian:
U:    No
```

Figure 9: An example of the “I don’t know” feature.

<u>Things it can do and things it could do:</u>

Currently the system can recommend one or more of the eight wines stored based primarily on a meal’s ethnicity and some user defined preferences. Future features to the program could include allowing the user to specify what type of wine they wanted, with the program outputting a recipe that would be compatible. Currently, the program can’t handle starting with a wine and recommending a recipe. Another adjustment that would allow the program to go beyond its current task would be to incorporate more specific wines by utilizing production years, production origins and location of availability. Thus, instead of simply outputting a type of wine, the program could recommend a year if one was particularly better than the others. This could easily be accomplished by adding rules for more specific wines.

Another feature that our system could be modified to include in the future is the recommendation of a wine for a group of people that are eating together. Currently our system assumes that the recommendation of a wine is for one member of the party. If multiple people in a table share similar meals or tastes then we could easily add more rules that would output some common wines that would fit for all members of the table.

Additionally one other modification that could be done to the system is the addition of a case-based reasoning paradigm. With this, the system could remember previous recommendations and then recommend it to a new user if he/she has a similar meal. The user might tell the system which wine he finally chose and if he enjoyed or disliked the combination.

What does it know?

The system utilizes a backward-chaining rule-based system as its knowledge representation. Due to the size of the graphic, a copy of the overall view of the tree structure of all the rules that contribute to the final goal of “wine-to-drink” in our system will not be presented here, but will be available with this file (called rule structure.png)

In order for our system to be more than just a simple database lookup application we used several indirect attributes to obtain the necessary information to conclude the most appropriate wines for a specific meal. The biggest and most important attribute is the meal ethnicity. With this information we can obtain a clear idea of the possible ingredients a meal might have, ranging from the most common herbs and spices that could be used, to the types of meat that a user is most likely to eat. Following this, the system looks for some other attributes that relate more to the user’s preference to narrow down the possible wine recommendations. Most of these preferences, if not all can be determined by two different ways: either by answering the question directly or, if given the case that the user is not sure or does not know, by answering some other more simple questions that conclude these preferences.

How does it work?

The system works similarly to the abovementioned worked out example. The program begins by asking the user what type of ethnicity is the meal he/she is expected to eat. The answer to this question provides many details to the possible meal. With the food’s ethnicity we are able to determine, the type herbs, sauces and meats that might be available in a meal. After this, we then proceed to ask what type of meal will the user consume, this could be lunch, dinner or dessert. If it’s a dessert then the system will automatically realize that the appropriate wine is Riesling, since it’s the only one in our system that corresponds to a dessert meal. Answering lunch or dinner will result in more questions to infer the type of meat in the meal. Once the system knows what meat will be used, it will then ask some questions that correspond to that particular ethnicity to further narrow the possible choices of wines. Finally, once it obtains all the necessary information the system then outputs a list of all the wines that it could recommend along with a certainty factor that is determined by the amount of attributes that match that given wine.

What went well?

Overall, our system was successful; it satisfactorily executed the proposed task of recommending a wine to a user depending on the type of meal to be consumed. Our system had some nice advantages. For starters, the system is completely modular. If needed be, a person could add or remove ethnicities or second level knowledge without much effort. This makes the system easily extensible and modifiable. Along with this, the use of domain or ethnicity specific questions facilitated the narrowing down of the different possible wine recommendations.

What didn't go so well?

Along with some advantages, our system also had some issues that should be addressed. One of our main issues resulted from the use of Joshua and rules as our knowledge representation system. Each wine had a large number of attributes that could be used for its recommendation. In many situations we had to create rules for many, if not all, the possible combination of attributes such that all the different meal combinations would return at least one wine. In one case this resulted in 26 rules for a specific wine. The advantage to this was that we could use these permutations as a way to determine the certainty that a meal goes well with that wine. You can observe by looking at the rules for wines in the appendix.

Another issue that I consider as a disadvantage to our system is that we had to do some broad generalizations and assumptions in order simplify the task. I believe that some of these overgeneralizations cause some a loss in the accuracy of the results. This tradeoffs should be further studied analyzed to see where we could improve the accuracy of the results without overcomplicating the task.

Finally, I would have liked if the final output of the system would result in a more general, natural language style explanation of the recommendations along the reasoning used behind it. As it stands right now, the explanation only returns a list of all the rules used in obtaining the result in Joshua format. I would have also liked if along each of the recommendations, the user might obtain a short blurb with more information on the particular wine in order to make a more informed decision

What did you learn from this?

During the course of this project I learned various things. For instance, I learned that a recommendation is only as good and effective as the amount of information that is given to you by a user. If the user does not offer you with the necessary information then the quality and accuracy of your recommendation will not be as high as expected. More of the recommendations would be more like educated guesses than recommendations based on valid facts. This also goes along with assumptions and generalizations. These, while good for simplifying a task, can eventually lower the accuracy and truthfulness of the final result to the point that it might not be entirely useful. On the other hand, its quite interesting on the type of information that a person might obtain or consider as commons sense without asking the user directly.

In conclusion we believed that the use of rules might not have been the best choice, for this task. Frames could have been better suited for this. If we had worked with frames, some issued such as the multiple rules due to different permutations could have been avoided and the end product could have been more efficient and simpler/

Finally this task is one that can be difficult and requires time and experience in order to fully master. The system we offered a good start in achieving this, but it still has much room for improvement in terms of accuracy and usability. Overall, this was a fun project and challenge to do and demonstrated the difficulty of knowledge engineering.

APPENDIX

The following are some example rules for an ethnicity and for a specific wine type:

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;                                                         ;;;
;;;  Inference Rules (For importance, higher values go first.)  ;;;
;;;  Domain:  American Food   Specific                      ;;;
;;;                                                         ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; Meal Type

(defrule full-meal-american (:backward :certainty 1.0 :importance 91)
  if [and          [food-ethnicity ?user american]
                             [or [meal ?user lunch]
                                 [meal ?user dinner]]]
  then [full-meal-american ?user yes])

(defrule dessert-meal-american (:backward :certainty 1.0 :importance 91)
  if [and          [food-ethnicity ?user american]
                             [meal ?user dessert]]
  then    [dessert-meal-american ?user yes])

(defrule full-meal-american-exclusion (:forward :certainty 1.0 :importance 90)
          if [full-meal-american ?user yes]
  then [and [dessert-meal-american ?user no]
            [having-dessert ?user no]])

(defrule dessert-meal-american-exclusion (:backward :certainty 1.0 :importance 91)
  if [and          [food-ethnicity ?user american]
                   [meal ?user dessert]]
  then [full-meal-american ?user no])

          ;;; Figure out meat

                    ;;; General selection

(defrule meat-type-healthy-american (:backward :certainty 1.0 :importance 96)
  if [and          [food-ethnicity ?user american]
                   [full-meal-american ?user yes]
                   [health-conscious ?user yes]]
  then [meat-american-round1 ?user poultry/seafood])

(defrule meat-type-not-healthy-american (:backward :certainty 1.0 :importance 95)
  if [and          [food-ethnicity ?user american]
                   [full-meal-american ?user yes]
                   [health-conscious ?user no]]
  then [meat-american-round1 ?user pork/beef])

                    ;;; final selection

 (defrule bbq-american (:backward :certainty 1.0 :importance 94)
   if [and [food-ethnicity ?user american]
          [full-meal-american ?user yes]
          [meat-american-round1 ?user pork/beef]
          [is-south ?user yes]
          [is-summer ?user yes]
          [use-bbq-sauce ?user yes]]
   then   [final-meat ?user BBQ])

 (defrule beef-american (:backward :certainty 1.0 :importance 93)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user pork/beef]
                             [is-midwest ?user yes]]
   then [beef ?user yes])

  (defrule beef-american-no (:backward :certainty 1.0 :importance 93)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user pork/beef]
                             [is-midwest ?user no]]
   then [beef ?user no])

  (defrule beef-american-no-2 (:backward :certainty 1.0 :importance 93)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user poultry/seafood]]
   then [beef ?user no])
 (defrule seafood-american (:backward :certainty 1.0 :importance 92)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user poultry/seafood]
                             [is-near-coast ?user yes]]
   then [seafood ?user yes])

  (defrule seafood-american-no (:backward :certainty 1.0 :importance 92)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user poultry/seafood]
                             [is-near-coast ?user no]]
   then [seafood ?user no])

  (defrule seafood-american-no2 (:backward :certainty 1.0 :importance 92)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user pork/beef]]
   then [seafood ?user no])

 (defrule poultry-american (:backward :certainty 1.0 :importance 91)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user poultry/seafood]
                             [is-near-coast ?user no]]
   then [poultry ?user yes])

  (defrule poultry-american-no (:backward :certainty 1.0 :importance 91)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user poultry/seafood]
                             [is-near-coast ?user yes]]
   then [poultry ?user no])

  (defrule poultry-american-no2 (:backward :certainty 1.0 :importance 91)
   if [and         [food-ethnicity ?user american]
                             [full-meal-american ?user yes]
                             [meat-american-round1 ?user pork/beef]]
   then [poultry ?user no])

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;                                                        ;;;
;;; Riesling                                               ;;;
;;;                                                        ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defrule riesling-desert (:backward :certainty 0.9 :importance 90)
          if        [meal ?user dessert]
          then [wine-to-drink ?user riesling])

(defrule riesling-desert-2 (:backward :certainty 1.0 :importance 91)
          if        [and      [meal ?user dessert]
                                        [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])

(defrule riesling-1 (:backward :certainty 1.0 :importance 92)
          if        [and      [or [poultry ?user yes]
                                                  [shellfish ?user yes]]
                                        [has-cheese ?user yes]
                                        [use-light-sauce ?user yes]
                                        [or       [use-dill ?user yes]
                                                  [use-sage ?user yes]
                                                  [use-clove ?user yes]
                                                  [use-ginger ?user yes]]
                                        [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])

(defrule riesling-2 (:backward :certainty 0.8 :importance 83)
          if        [and      [or [poultry ?user yes]
                                                  [shellfish ?user yes]]
                                        [use-light-sauce ?user yes]
                                        [or       [use-dill ?user yes]
                                                  [use-sage ?user yes]
                                                  [use-clove ?user yes]
                                                  [use-ginger ?user yes]]
                                        [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])
```

```
(defrule riesling-3 (:backward :certainty 0.8 :importance 83)
          if        [and      [or [poultry ?user yes]
                                        [shellfish ?user yes]]
                              [has-cheese ?user yes]
                              [or       [use-dill ?user yes]
                                        [use-sage ?user yes]
                                        [use-clove ?user yes]
                                        [use-ginger ?user yes]]
                              [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])

(defrule riesling-4 (:backward :certainty 0.5 :importance 83)
          if        [and      [or [poultry ?user yes]
                                        [shellfish ?user yes]]
                              [has-cheese ?user yes]
                              [use-light-sauce ?user yes]
                              [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])

(defrule riesling-5 (:backward :certainty 0.5 :importance 83)
          if        [and      [or [poultry ?user yes]
                                        [shellfish ?user yes]]
                              [or       [use-dill ?user yes]
                                        [use-sage ?user yes]
                                        [use-clove ?user yes]
                                        [use-ginger ?user yes]]
                              [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])

(defrule riesling-6 (:backward :certainty 0.3 :importance 83)
          if        [and      [or [poultry ?user yes]
                              [shellfish ?user yes]]
                              [use-light-sauce ?user yes]
                              [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])

(defrule riesling-7 (:backward :certainty 0.3 :importance 83)
          if        [and      [or [poultry ?user yes]
                              [shellfish ?user yes]]
                              [has-cheese ?user yes]
                              [prefers-sweet-wines ?user yes]]
          then [wine-to-drink ?user riesling])
```