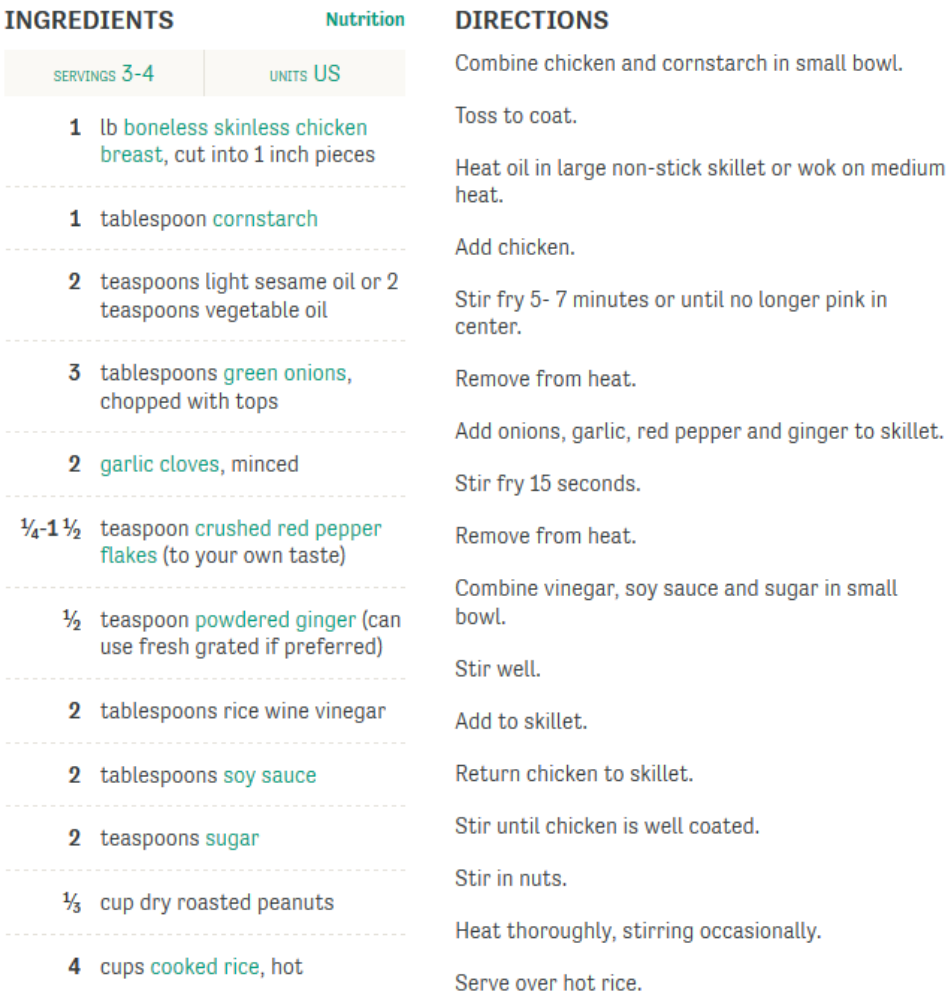
CS7637
Assignment 3

AI agent for designing new recipes using Frames

The goal is to create an AI agent that can design new recipes.

Let's take a look at the recipe of a classic Chinese Stir-fry: Kung Pao Chicken (Fig. 1). The left side, it lists all ingredients that used, e.g. sliced chicken breast, green onion, red pepper, et al. And the right side, it gives detailed instructions on how to cook the food, which starts from marinating chicken meat, preparing sauce and followed by stir-frying the ingredients on a hot wok. You will easily find out that a recipe is not mechanically mixing all ingredients, instead designing a new recipe is heavily knowledge based. The challenge of designing such an AI agent is how to pre-deposit all information that required for recipe designing. The information can be very broad, including properties of each raw materials, such as various meats, vegetables, seasonings, and the often used cooking procedures, like marinating, stir-frying, et al.

Figure 1



INGREDIENTS Nutrition

SERVINGS 3-4 | UNITS US

- 1 lb boneless skinless chicken breast, cut into 1 inch pieces
- 1 tablespoon cornstarch
- 2 teaspoons light sesame oil or 2 teaspoons vegetable oil
- 3 tablespoons green onions, chopped with tops
- 2 garlic cloves, minced
- ¼-1 ½ teaspoon crushed red pepper flakes (to your own taste)
- ½ teaspoon powdered ginger (can use fresh grated if preferred)
- 2 tablespoons rice wine vinegar
- 2 tablespoons soy sauce
- 2 teaspoons sugar
- ⅓ cup dry roasted peanuts
- 4 cups cooked rice, hot

DIRECTIONS

Combine chicken and cornstarch in small bowl.

Toss to coat.

Heat oil in large non-stick skillet or wok on medium heat.

Add chicken.

Stir fry 5- 7 minutes or until no longer pink in center.

Remove from heat.

Add onions, garlic, red pepper and ginger to skillet.

Stir fry 15 seconds.

Remove from heat.

Combine vinegar, soy sauce and sugar in small bowl.

Stir well.

Add to skillet.

Return chicken to skillet.

Stir until chicken is well coated.

Stir in nuts.

Heat thoroughly, stirring occasionally.

Serve over hot rice.

A Frame is an artificial intelligence data structure that used to store or represent knowledge. It use slots to store values. The information on recipe of Kung Pao Chicken can be stored as a frame (Table 1). It contains frame name: “Kung Pao Chicken”, and AKO (a kind of) defines its category, that it is one kind of “Chinese Stir-fry”, and many slots name as Serves, Ingredient A, B, C… with specific values filled. For some slots it also has default values, e.g. procedure 1: marinate, procedure 2: stir-fry, Hot plate : true, that is because a traditional “Chinese Stir-fry” always includes marinating and stir-frying procedures and always serves as hot plate.

Table 1

Slot	Value	Type
Kung Pao Chicken	-	(This Frame)
AKO	**Chinese Stir-fry**	(parent frame)
Serves	4	(instance value)
Time cost	40 min	(instance value)
Ingredient A	Chicken breast	(instance value)
Amount	1 lb	
Ingredient B	Green Onion	(instance value)
Amount	0.5 lb	
Ingredient C	Red pepper	(instance value)
Amount	50 gram	
.....	……	(instance value)
Procedure 1	**Marinate**	(default value)
Time	15 min	(instance value)
Procedure 2	**Stir-fry**	(default value)
Time	10 min	(instance value)
……	……	(instance value)
Hot plate	true	(default value)

However, this frame is not created by AI agent, instead it is directly transformed from available information (Fig. 1). To generate a new frame (recipe) like this, the agent need to know more general information, for example, the composition of a general Chinese Stir-fry recipe, which is the parent frame of “Kung Pao Chicken” frame. The parent frame, named “Chinese Stir-fry”, is shown in Table 2. In the frame, “Chinese Stir-fry” is the frame name, and since it belongs to “Culinary Dish”, a frame named “Culinary Dish” will be its parent frame. In the “Chinese Stir-fry” frame, the slot value of Serves can be changed, and the time cost also varies based on its ingredients. For the Ingredient A, instead of a specific (instance) value: chicken breast, it is more general, that any kind of meat will be used, the amount of meat will be calculated based on Serves value. Same

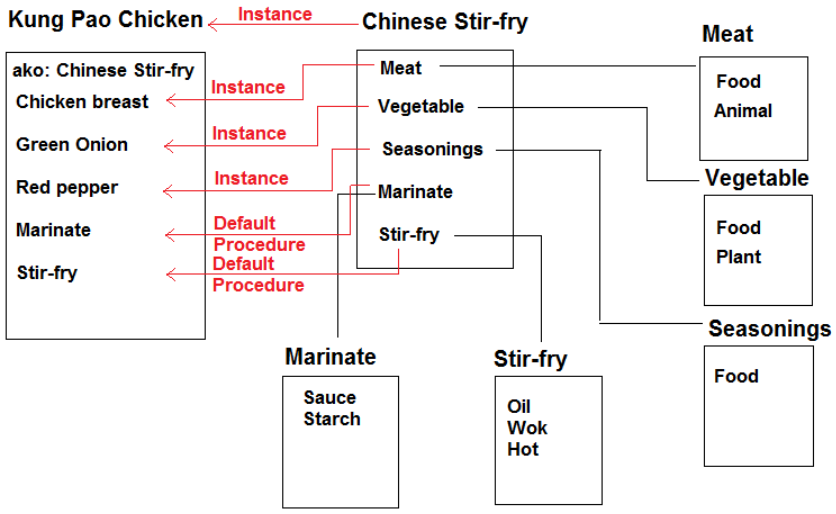
thing happens for Ingredient B, as any kind of vegetables, and Ingredient C, as any kind of seasonings. The frame “Chinese Stir-fry” has two default procedures, that Procedure 1: Marinate and Procedure 2: Stir-fry, and also set Hot plate as true for default. There are many other parent frames listed in this frame. For example, the “Meat” frame, describing that meat is kind of food (parent frame), and it comes from animals (parent frame), it also contains various nutrients, and usually cooked before serving (default value). The frame “Stir-fry” will list utensils that used in stir-fry procedure, e.g. a wok, and other materials, e.g. oil. I will not list all details here, since that will be very complicated.

Table 2

Slot	value	Type
Chinese Stir-fry	-	(This Frame)
AKO	**Culinary Dish**	(parent frame)
Serves	2 (default) IF-ADDED:	
Time cost	30 min (default) IF-ADDED:	
Ingredient A	**Meat**	(parent frame)
Amount	0.5 lb (default) IF-ADDED:	Calculation based on Serves procedure attached
Ingredient B	**Vegetable**	(parent frame)
Amount	0.2 lb (default) IF-ADDED:	Calculation based on Serves procedure attached
Ingredient C	**Seasonings**	(parent frame)
Amount	IF-ADDED:	procedure attached
.....	……	
Procedure 1	**Marinate**	(procedure frame)
Time	IF-NEEDED:	Calculation based on Serves procedure attached
Procedure 2	**Stir-fry**	(procedure frame)
Time	IF-NEEDED:	Calculation based on Serves procedure attached
……	……	
Hot plate	true	(default value)

The relations between these frames are further illustrated in Fig 2. Since the similarity between Frames and Object-Oriented Programming, I even can write some python codes to further implement the ideas.

Figure 2




```
class Chinese_Stir_fry(Culinary_Dish):      # Inherit from Culinary_Dish class
    def__init__(self, meat, vegetables, seasonings, serves = 2):
    ……
    # parameter meat should be a meat object, e.g. “chicken breast” or “beef” …
    # parameter vegetables should be a list of vegetable objects, e.g. [“green onion”, “asparagus”, “mushroom”, …]
    # parameter seasonings should be a list of seasoning objects, e.g. [“red pepper”, “brown sugar”, “vinegar”…]
    # parameter serves should be an integer with a default value 2.
    def calculateAmount(self):     # calculate each ingredients amount based on serves.
    ……
    def makeSauce(self):
    ……
    # calculate seasonings’ amounts based on serves and the choice of meat and vegetables.
   def calculateMarinateTime(self):
    ……
   # calculate marinate time based on the choice of meat.
   def calculateStirfryTime(self):
   ……
   # calculate stir-fry time based on the choice of meat and vegetables.
   def calculateTimeCost(self):
   ……
   # calculate time cost based on marinate time and stir-fry time, and extra food preparation time.
   def printRecipe(self):              #print out the generated recipe.
   ……
```

Finally, let's assume that such an agent has been created, and let me show you how it works. Step 1, the agent will run a search method to search food objects in refrigerator and return a list of food objects including various meats, vegetables and seasonings. They will be the starting materials for the new recipe. Step 2, the agent will run createRecipe method to create a new Chinese_Stir_fry object. You need to provide 1) preferred food list which has a default value: None, 2) serves which has a default value: 2. In the createRecipe method, the agent will first check the preferred food list that whether it contains any meat objects, vegetable objects or seasoning objects. If it is none, then randomly pick one meat object from food list, and pick 2-3 vegetable objects, and a few seasoning objects, then pass them to Chinese_Stir_fry class to create a new object. Step 3, printout the new recipe.

```
foodList = agent.search()

# return a list of food objects, e.g. ["chicken breast", "green onion", "sugar", "salt",…]

newRecipe = agent.createRecipe(["chicken breast", "red pepper"], 4)

# details about createRecipe method
# first check whether preferredFood == None, if not transform objects from the list.
# m = "chicken breast"
# v = [randomly pick vegetable objects from foodList]
# s = ["red pepper", randomly pick seasonings objects from foodList]
# create new recipe object: new_recipe = Chinese_Stir_fry(m, v, s, 4),
# return new_recipe

newRecipe.printRecipe()   # print out the new recipe
```