Repurposing of Resources: from Everyday Problem Solving to Crisis Management

A Leverhulme-funded research project in the Dept. of Computer Science and the Dept. of Information Studies at UCL

Anthony Hunter, Antonis Bikakis, Aissatou Diallo, Luke Dickens and Rob Miller

Duration: March 2022 - March 2025

{ KIDS }
Knowledge Information & Data Science

Motivation: repurposing is everywhere

The human ability to repurpose objects and processes is universal:

- in everyday situations, e.g. finding substitutes for missing cooking ingredients, or for unavailable tools for DIY.
- in professional life, e.g. clinicians often repurpose medicines off-license
- in addressing societal challenges, e.g. finding new roles for waste products
- in critical, unprecedented situations needing crisis management, e.g. to make shelter, distribute food, etc after natural disasters

Despite the importance of repurposing, the topic has received little academic attention.

Project aims

Develop a general theory or model of repurposing

- A classification scheme of types of repurposing
- A set of properties for characterising each type
- Methods for modelling underlying mechanisms for each type

Develop computational AI-based tools for repurposing

- Combination of methods and techniques from statistical and symbolic AI
- Machine Learning, Natural Language Processing, Word Embeddings, Ontologies, Knowledge Graphs, Commonsense Reasoning, Argumentation

Undertake case studies to evaluate theory and tools

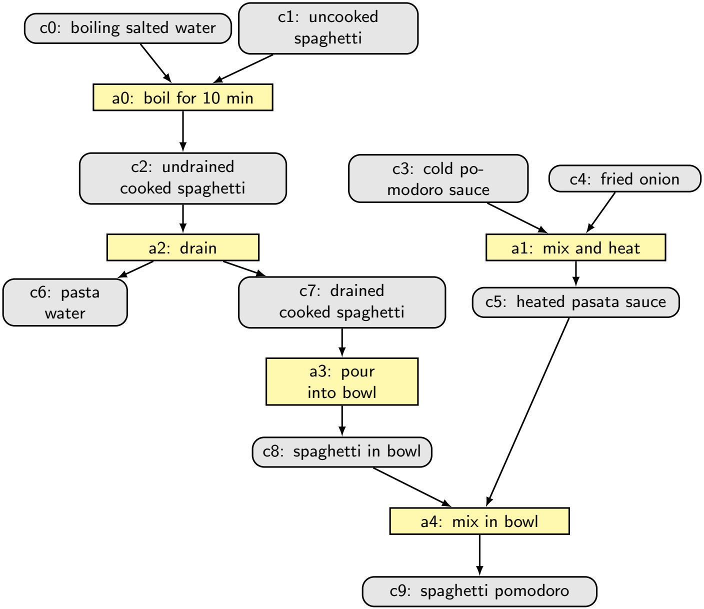
- Cooking, DIY, Recycling, Crisis management, Helping disadvantage people

Graphical formalism for human processes and substitution within them

For human processes, such as cooking recipes, that combine and transform resources into products and by-products:

- Represent as a bipartite directed graph with two node types: comestibles and actions
- Comestibles are input ingredients, intermediate items, products and by-products.
- Actions take one or more input comestibles, and output one or more transformed comestibles
- Both comestibles and actions might need to be subtituted for various reasons or situations

Example recipe graph



Some Formal Definitions

Definition

A **recipe graph** is a tuple (C, A, E) where: (1) $\emptyset \subset C \subseteq \mathcal{C}$ and $\emptyset \subset A \subseteq \mathcal{A}$; (2) E is a set of arcs that is a subset of $(C \times A) \cup (A \times C)$; (3) $(C \cup A, E)$ is a weakly connected directed acyclic graph; (4) for all $n_a \in A$, there are arcs (n_c, n_a) and (n_a, n'_c) in E; and (5) for all $n_c \in C$, if $(n_a, n_c), (n'_a, n_c) \in E$, then $n_a = n'_a$.

Definition

A **recipe** R is a tuple (C, A, E, F) where (C, A, E) is a recipe graph and $F : C \cup A \to \mathcal{T}_{\mathtt{C}} \cup \mathcal{T}_{\mathtt{A}}$ is a **typing function** that assigns a comestible (respectively action) type to each comestible (respectively action) node in C (respectively A) s.t. for all $n, n' \in C$, if $F(n) \simeq F(n')$, then $n = n'$. For a recipe $R = (C, A, E, F)$, we also define the following notation: $\text{Graph}(R) = (C, A, E)$, $\text{Type}(R) = F$, $\text{Nodes}(R) = C \cup A$, and $\text{Arcs}(R) = E$.

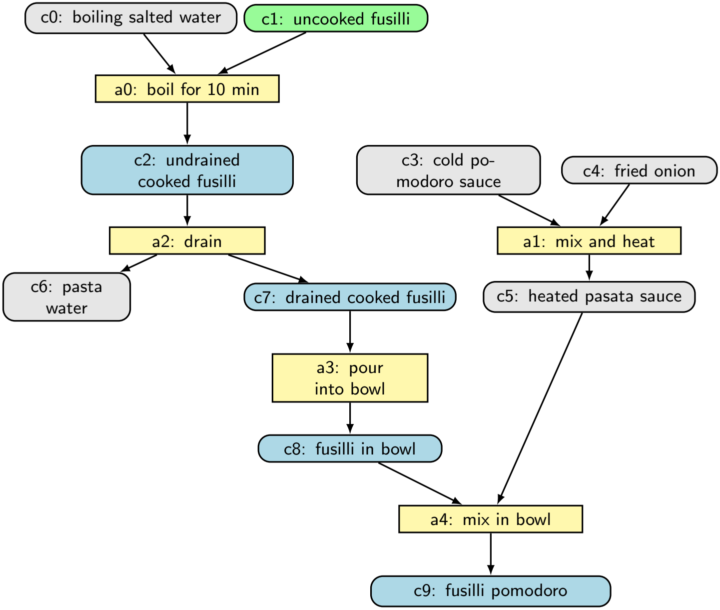
The ASP code for a recipe graph

```
% C contains only comestible nodes:
:- in(c_nodes(RG),X), not is_c_node(X), recipe_graph(RG).
% A contains only action nodes:
:- in(a_nodes(RG),X), not is_a_node(X), recipe_graph(RG).
% E contains only arcs:
:- in(arcs(RG),X), not is_arc(X), recipe_graph(RG).

% Recipe graphs must have non empty comestible, action and arc sets
:- empty(c_nodes(RG)), recipe_graph(RG).
:- empty(a_nodes(RG)), recipe_graph(RG).
:- empty(arcs(RG)), recipe_graph(RG).

% Recipe graphs must not be cyclic
:- cyclic(RG), recipe_graph(RG).
% Recipe graphs must be connected
:- -connected(RG), recipe_graph(RG).
% Action nodes in recipe graphs must be properly connected
% (in and out going edges to comestible nodes)
:- -a_node_properly_connected(RG,a(N)), recipe_graph(RG).

% for each comestible node in a recipe graph, there is at
% most one incoming arc:
:- in(arcs(RG),arc(a(N1),c(N))), in(arcs(RG),arc(a(N2),c(N))),
   N1 != N2, recipe_graph(RG).
```

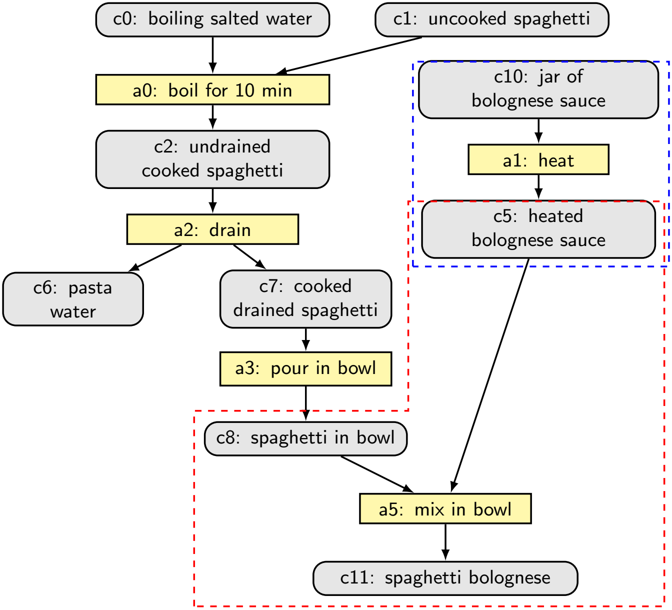
The ASP code for a recipe

```
% the 3rd argument of type_of is a function of the first two arguments:
% range of function is the c_nodes and a_nodes of corresponding graph
1 { type_of(TF,c(N),Ctype) : comestible_type(Ctype) } 1 :-
    recipe(RG,TF), in(c_nodes(RG),c(N)).
1 { type_of(TF,a(N),Atype) : action_type(Atype) } 1 :-
    recipe(RG,TF), in(a_nodes(RG),a(N)).

% for each recipe, type_of is restricted to its nodes only:
:- type_of(TF,N,T), recipe(RG,TF), not in(nodes(RG),N).

% all the comestible node types in a recipe must be in different type paths:
:- recipe(RG,TF), in(c_nodes(RG),N1), in(c_nodes(RG),N2), N1 != N2,
   type_of(TF,N1,T1), type_of(TF,N2,T2), same_type_path(T1,T2).
```

Type substitution



Structural substitution



What the ASP implementation can do

The code currently allows us to:

- define **type hierarchies** of actions and comestibles
- represent a corpus of **given recipes**
- extract a collection of **acceptability tuples** from the given recipes
- test the **validity** of **candidate recipes** with respect to the acceptability tuples
- suggest **type substitutions** for recipes, using information in the type hierarchies
- given a **primary substitution** within a recipe, find a corresponding set of **secondary substitutions** that will re-validate the recipe at **minimal cost**

Other Ongoing Work Within The Project

- **PizzaCommonSense** - a crowdsourcing annotated corpus of recipes for implicit intermediate commestibles
- Step by-step reasoning framework and associated prompt engineering for improved LLM output
- Commonsense reasoning framework for substitution, with improved recipe similarity measures, and mitigation capability
- Crowdsourcing crisis management responses to interrogate reasoning capabilities of LLMs
- **RecipeAnalysis** python library and database - extracting knowledge from recipe datasets to build structured representations at large scale
- **recipe2graph** unsupervised graph extraction from text recipes

Crowdsourcing crisis management responses

Context: This situation depicts a roadway with vehicles driving by a significant fire. The fire appears to be burning intensely, emitting bright flames and a lot of smoke. The scene looks like it might be taking place near a wooded or residential area, as trees and structures are visible in the vicinity.

Problem: heat and smoke causing an inhalation hazard to people in the area

Missing resource: clean air

Question: If a wildfire occurs near a highway, what **immediate** action is necessary to protect people from inhaling smoke?

Answer: close roads going into the area to prevent access and mobilize firefighters

Explanation: this reduces the number of people impacted by smoke inhalation by removing them from the hazard and removes hazard itself.