A Scalable Combinatorial Optimization Solution to ICT Operational Problems through Technology Adoptions

September 3, 2020

1 Executive Summary

Whereas human minds can only feasibly compare a limited number of combinations with limited variables / dimensions, computers can compare n combinations in k-dimensions, enabling humans to solve large, complex problems. A computer can run through all combinations within a dataset to find a configuration that best fits a given critera, using a process known as combinatorial optimization. In fact, such processes are the building blocks of machine learning (ML) algorithms.

2 Theory

The formulation below describes a scalable means of finding optimal solutions to problems solved via the adoption of a configuration of technologies. For example, problems such as, "What collection of tools and/or processes can I use to best improve peer-to-peer communications in my organization given certain constraints?" can be solved with a selection of technologies that maximize utility with given datasets of technologies. I.e. The formulation provides a combination of technologies that best fits a given criteria.

The algorithm works as follows:

1. Given two datasets of technologies and their corresponding utility metrics (measurements of the degree to which a technology solves a defined problem), combine both datasets into one technologies dataset in which existing tech solutions are weighted higher than non-adopted tech solutions.
2. Given a list of constraints, return a set of configurations of technologies that satsify the constraints (AKA a feasible set).
3. Sort the feasible set from greatest utility to least utility and return the top n configurations.
4. Return the configuration within the top n that requires the least number of additional tools to adopt.

The algorithm can find optimal solutions to problems from levels as high as, "What technologies best facilitate collaboration in my organization?" to levels as low as, "Which email client should I use?"

The formulation is a modification of the combinatorial optimization *0/1 Knapsack Problem* and is solved using dynamic programming (DP) and memoization.

3 Problem Formulation

$$\text{optimal solution} = \min\ \text{sort}(A, n) \tag{1}$$

Where:

$$\text{sort} := f : A, n \to y, \text{where } |y| \leq n, n \in \mathbb{N}, n \leq |I| \tag{2}$$

$$y := \text{list result of a first-element-keyed top-down sort of list-casted set } A \tag{3}$$

Given definitions:

$$A := \left(\sum_{i \in I} u_i x_i, |X_i|\right) \text{ s.t. } \sum_{i \in I} w_i x_i \leq W \wedge \sum_{i \in I} u_i x_i \geq 0 \wedge \hat{c} = \text{True}, \forall \hat{c} \in C \tag{4}$$

$$I := I_{\text{new}} \cap I_{\text{existing}} \tag{5}$$

$$I_{\text{new}} := \text{discrete set of potential tech solutions (e.g. tools, processes)} \tag{6}$$

$$I_{\text{existing}} := \text{discrete set of existing tech solutions (e.g. tools, processes)} \tag{7}$$

$$x_i := \begin{cases} x_{\text{new}}, & \text{if } i \in I_{\text{new}} \\ x_{\text{existing}}, & \text{if } i \in I_{\text{existing}} \end{cases} \tag{8}$$

$$x_{\text{new}} := \begin{cases} 1, & \text{if } i \text{ is adopted, where } i \in I_{\text{new}} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

$$x_{\text{existing}} := \begin{cases} 1.5, & \text{if } i \text{ is adopted, where } i \in I_{\text{existing}} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

$$X_i := \{x_i | x_i = 1\}, i \in I \tag{11}$$

$$u_i := \text{utility of } i \in I,\ u_i \in \mathbb{R}^+ \tag{12}$$

$$w_u := \text{cost of } i \in I, w_i \in \mathbb{R}^+ \tag{13}$$

$$W := \text{budget (cost upper-bound)} \tag{14}$$

$$C := \text{discrete set of constraints} \tag{15}$$

4 Implementation

4.1 Usage Process

The process of optimizing a tech-adoption-enabled solution works as follows:

1. Define a problem that can be solved through the adoption of a configuration of tools and/or processes
2. Define a list of constraints on potential solutions (e.g. a budget constraint)
3. Define metrics for evaluating potential solutions to the problem
4. Compile a dataset of adopted technologies and their corresponding metric values
5. Compile a dataset of researched and non-adopted technologies and their corresponding metric values

6. Pass both datasets into the formula
7. Inform decisions through analyzing the results

Main Benefits:

- Solutions address root problems, instead of patching holes
- Solutions are evidence- and data-driven
- The problem solving process considers existing technologies as potential solutions and minimizes the need for adopting additional technologies
- The process and algorithm provide transparency and accountability in decision-making
- The process handles more data points and calculations than manually-feasible
- The process provides a methodological and quality standard in research and decision-making
- If used enough times (i.e. having built a training dataset), the algorithm can be adapted into a ML algorithm

Main Limitations:

- The formulation addresses problems that can be solved through the adoption of technologies (e.g. cannot solve cultural problems)
- The marginal utility of the process increases with data and analysis and thus is limited by time and labor constraint
- The formulation requires relevant and consistently defined utility metrics
- The evaluation of metrics are often subjective – their validity depends on data sourced from a representative sample
- Quantitative evaluations are subject to data biases and are limited to quantifiable measures
- There exists a risk of relying on the formulation; it serves to inform, not decide

4.1.1 Data Collection Needs

- $I_{existing}$: the current set of existing tools that contribute to solving the problem
- I_{new}: the current set of existing tools being researched for potential adoption
- W: budget limitations
- C: additional constraints
- u_i: utility measure of technology i
- w_i: cost measure of technology i

4.2 Example

4.2.1 Problem Formulation & Data Collection

N.B. The problem formulation and data below are hypothetical

Problem Statement: What is the best selection of tools for improving P2P communications in my organization?

Defined Constraints

- Budget: \$8,000 / yr
- No duplicate tools
- Minimum security level: MED

Utility Metrics

- Speed of communications
- Facilitates real-time conversations
- Intuitive interface
- Accessible
- Secure
- Platform/device/location agnostic

Dataset of Adopted Technologies

	Communications Speed	Synchronous	Intuitive Use	Accessible	Secure	Agnostic	Cost / yr
mail	LOW	LOW	LOW	MED	LOW	LOW	\$2000
desk phone	MED	HIGH	MED	MED	MED	LOW	\$4000
email 1	HIGH	MED	MED	HIGH	MED	HIGH	\$2000
email 2	HIGH	MED	HIGH	HIGH	MED	HIGH	\$3000

Dataset of Researched Technologies

	Communications Speed	Synchronous	Intuitive Use	Accessible	Secure	Agnostic	Cost / yr
IM Service 1	HIGH	HIGH	HIGH	HIGH	MED	MED	\$2000
IM Service 2	HIGH	HIGH	HIGH	HIGH	HIGH	HIGH	\$2500
Video Chat 1	HIGH	HIGH	HIGH	HIGH	MED	MED	\$3000
Video Chat 2	HIGH	HIGH	MED	HIGH	HIGH	MED	\$3000
Cell Phone	HIGH	HIGH	HIGH	HIGH	MED	HIGH	\$4000

4.2.2 Solution

Programming Solution using DP

Let $u_i = \frac{\sum \text{metrics}_i}{|\text{metrics}_i|}$, $\text{HIGH} = 3,\ \text{MED} = 2,\ \text{LOW} = 1$

[1]:
```
# TODO: In progress -- DP + memoization implementation
```