Mathematical Formulation of the AYTO Solver

Constraint Satisfaction & Surjective Bipartite Matching

This document outlines the mathematical foundation of the parallelized constraint satisfaction solver designed for the logic puzzle from the reality television show **Are You The One?** (AYTO).

1. Variables and Sets

Let G_1 be the first group (e.g., group 1 candidates) of size N, and G_2 be the second group (e.g., group 2 candidates) of size M, where $N \leq M$.

The algorithm seeks to find all valid mappings between the two groups. Mathematically, it searches for all valid **surjective functions** $f : G_2 \to G_1$. Surjectivity implies that every person in G_1 must be matched with at least one person in G_2.

2. Constraints

The function f must satisfy two primary types of constraints:

2.1. Allowed Mappings (Truth Booths)

Let $A(y) \subseteq G_1$ represent the set of allowed matches for a given $y \in G_2$. Initially, $A(y) = G_1$ for all y. The "Truth Booths" apply absolute constraints:

- **Confirmed Match (True):** If (x, y) is a confirmed match, then $A(y) = \{x\}$.
- **Confirmed Non-Match (False):** If (x, y) is a confirmed non-match, then $A(y) = A(y) \setminus \{x\}$.

In the implementation, $A(y)$ is efficiently represented as an N-bit integer, where the i-th bit is 1 if $x_i \in A(y)$, and 0 otherwise.

2.2. Ceremony Constraints

Let C be the set of all ceremonies. Each ceremony $c \in C$ consists of a set of guessed pairs $P_c \subset G_1 \times G_2$ and a score $b_c \in \mathbb{N}$ (the "beams"), representing the exact number of correct matches in that guess.

For every ceremony $c \in C$, a valid function f must satisfy:

$$\sum_{(x,y)\in P_c} \mathbb{I}(f(y) = x) = b_c$$

where $\mathbb{I}$ is the indicator function that equals 1 if the condition is true and 0 otherwise.

3. Search Space and Pruning

The algorithm explores the search space using a Depth-First Search (DFS) tree, building the function f sequentially. Let f_k be a partial assignment of the first k elements of G_2.

Pruning by the Pigeonhole Principle:

At depth k, let $U_k \subseteq G_1$ be the set of elements in G_1 that have already been assigned a match.

- The number of elements in G_1 still needing an assignment is $N - |U_k|$.
- The number of unassigned elements in G_2 is $M - k$.

Because the final function f must be surjective, if the number of required assignments is greater than the number of available slots, the branch is physically impossible and is pruned immediately:

$N - |U_k| > M - k \Rightarrow \text{Prune branch}$

4. Objective

Let $\mathcal{F}$ be the set of all fully assigned, valid functions f that survive the pruning and satisfy all ceremony constraints.

The algorithm outputs two standard metrics:

1. **Total Possibilities:** $|\mathcal{F}|$
2. **Marginal Probabilities:** A matrix $P \in [0,1]^{N \times M}$, representing the likelihood of each specific pairing across all valid universes.

For each $x_i \in G_1$ and $y_j \in G_2$, the probability is calculated as the ratio of valid functions that contain that specific match over the total number of valid functions:

$P_{i,j} = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \mathbb{I}(f(y_j) = x_i)$