Moving to a different formalism...

SEND
+ MORE
------
MONEY

Consider search space for cryptarithmetic.
DFS (depth-first search)
Is this (DFS) how humans tackle the problem?
And if not, what do humans do?

Human problem solving appears much more sophisticated!
For example, we derive new constraints on the fly.
In a sense, we try to solve problems with little
or no search!
In example, we can immediately derive that $M = 1$.
It then follows that $S = 8$ or $S = 9$. Etc. (derive more!)

Constraint Satisfaction Problems (CSP)

A powerful representation for (discrete) search problems.
A **Constraint Satisfaction Problem (CSP)** is defined by:
- $X$ is a set of $n$ variables $X_1, X_2, \ldots, X_n$,
each defined by a finite domain $D_1, D_2, \ldots, D_n$
of possible values.
- $C$ is a set of constraints $C_1, C_2, \ldots, C_m$.
  Each $C_i$ involves some subset of the variables; specifies
  the allowable combinations of values for that subset.
A **solution** is an assignment of values to the variables
that satisfies all constraints.

Cryptarithmetic as a CSP

TWO
+ TWO
------
FOUR
Variables:
\[ T = \{0, \ldots, 9\}; \ W = \{0, \ldots, 9\}; \ O = \{0, \ldots, 9\}; \]
\[ F = \{0, \ldots, 9\}; \ U = \{0, \ldots, 9\}; \ R = \{0, \ldots, 9\}; \]

Constraints:
\[ O + O = R + 10 \cdot X_1 \]
\[ X_1 + W + W = U + 10 \cdot X_2 \]
\[ X_2 + T + T = O + 10 \cdot X_3 \]
\[ X_3 = F \]
\[ \text{TWO} + \text{TWO} = \text{FOUR}; \]

each letter has a different digit (\( F \neq T, F \neq U, \text{etc} \));

**Cryptarithmetic Constraint Graph**

**Constraint Satisfaction Problems (CSP)**

For a given CSP the problem is one of the following:

- find all solutions
- find one solution
- just a feasible solution, or
- a “reasonably good” feasible solution, or
- the optimal solution given an objective function
- determine if a solution exists
How to View a CSP as a Search Problem?

**Initial State** – state in which all the variables are unassigned.

**Successor function** – assign a value to a variable from a set of possible values.

**Goal test** – check if all the variables are assigned and all the constraints are satisfied.

**Path cost** – assumes constant cost for each step.

---

**Branching Factor**

**Hypothesis 1** – any unassigned variable at a given state can be assigned a value by an operator: branching factor as high as sum of size of all domains.

**Better approach** – since order of variable assignment not relevant, consider as the successors of a node just the different values of a single unassigned variable: max branching factor = max size of domain.

**Maximum Depth of Search Tree**

$n$ the number of variables; all the solutions are at depth $n$.

What are the implications in terms of using DFS vs. BFS?

---

**CSP – Goal Decomposed into Constraints**

How to exploit it?

**Backtracking search**: a DFS that chooses values for variables one at a time, checking for consistency with the constraints.

An uninformed search.
Constraint propagation: “looking ahead”

- **Forward Checking** — each time variable is instantiated, delete from domains of the uninstantiated variables all of those values that conflict with current variable assignment.
- **Arc Consistency** — state is arc-consistent, if every variable has some value that is consistent with each of its constraints (consider pairs of variables)
- **K-Consistency** generalizes arc-consistency. Consistency of groups of K variables.

---

Variable and value ordering:

- **Minimum remaining values (MRV)**: choose the variable with the fewest possible values.

**Degree heuristic**: assign a value to the variable that is involved in the largest number of constraints on other unassigned variables.

**Least-constraining value heuristic**: choose a value that rules out the smallest number of values in variables connected to the current variable by constraints.
Dramatic recent progress in Constraint Satisfaction.

For example, methods can now handle problems with 10,000 to 100,000 variables, and up to 1,000,000 constraints.