Turing machine wrt Domains

Start with

X Domain types
Y ~~States~~ Instances w/in the X Domains — State

A Additional domains to represent coupling between the X Domains

B Additional states ~~to~~ within the Y Domains to represent the coupling state

Then you can model the system

Key observations

1) A/B are added state domains that store the coupling between domains

2) - X = 3 in turing machines
   - X > 3 in real observed world
   - turing machine can execute any computation

∴ state table (Domain) + current state (value) can be stored or represented by 3 Domains + States

In other words arbitrary number of additional Domains and states can be represented in turing machine

3) the turing machine serves to both

A) store additional states ~~and~~ in state domains within the 3 domains and states

B) Represent the relationships and coupling between the states

Observation
turing machine has

master state domain with
∞ states with a limited
set of transition types
allowed between states

Tape - data structure w / ∞ nodes
but ~~fixed # of states in nodes~~
fixed transitions between states

Chars - fixed states and transitions
but ∞ instances

Instances States Transitions

Tape Cells Symbol

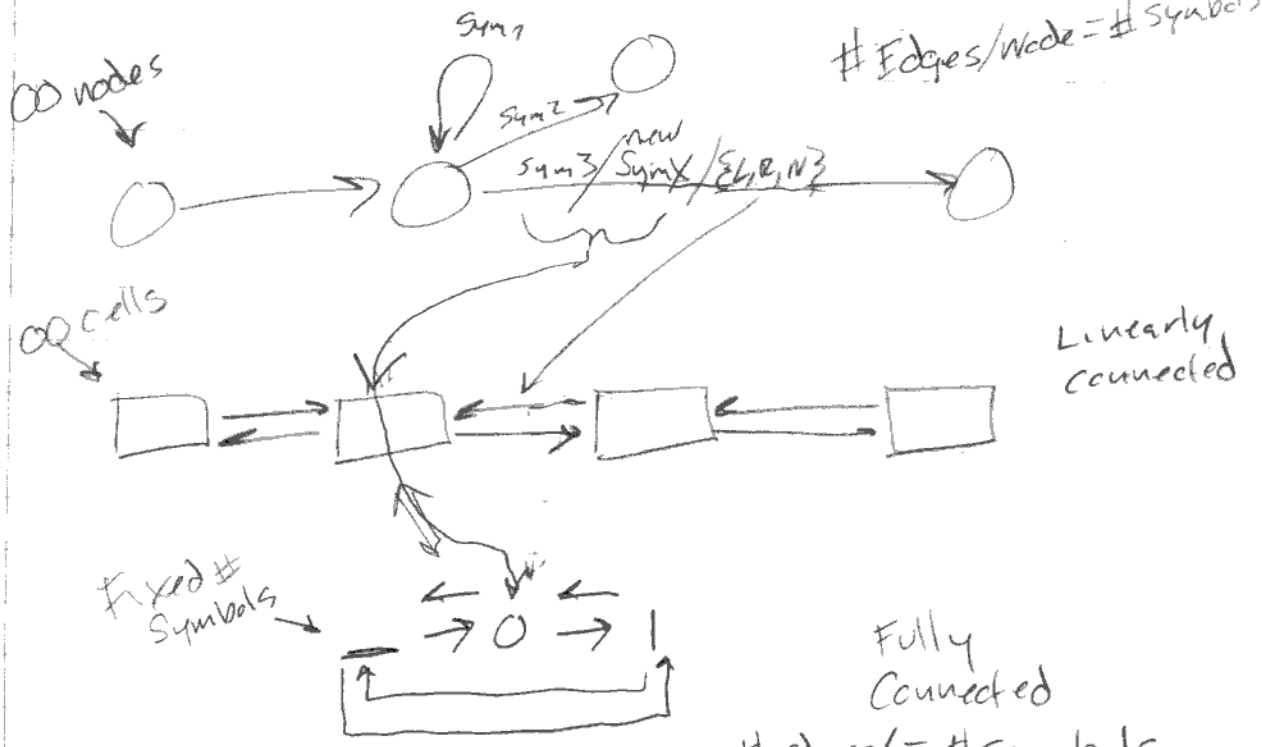
blank symbol – '0'

Head

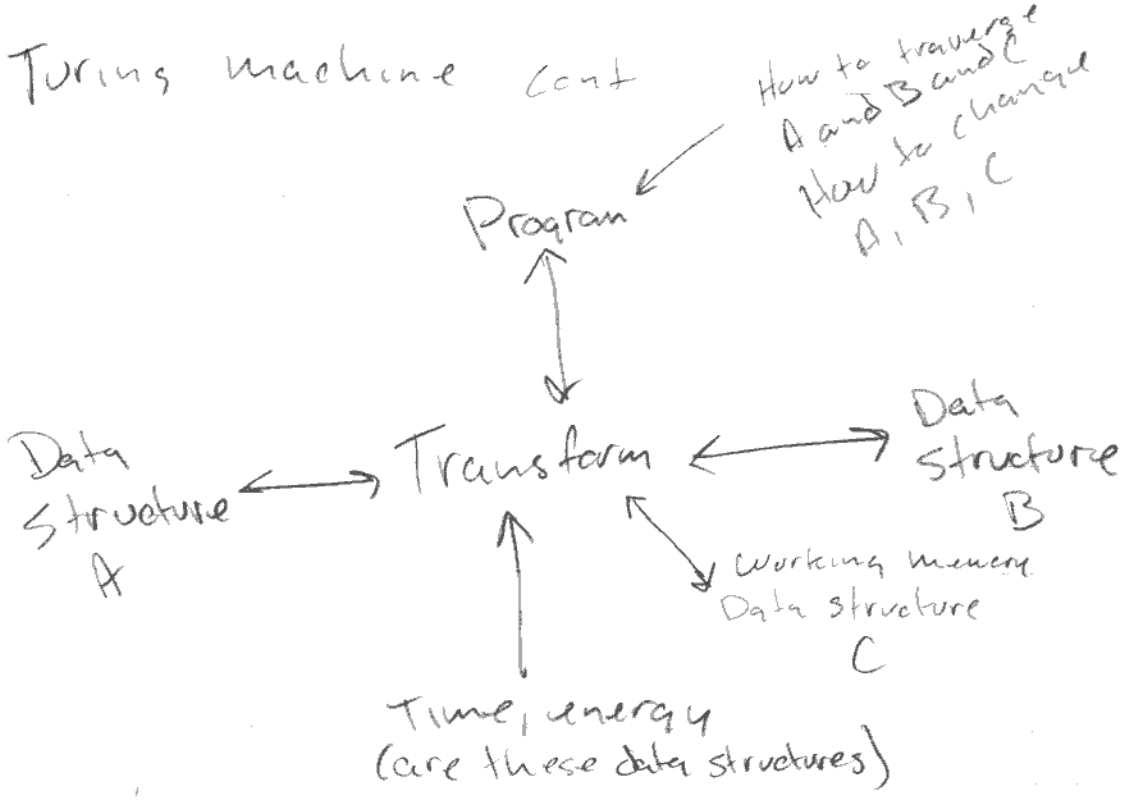
State register

instruction table

Movement L, R, N

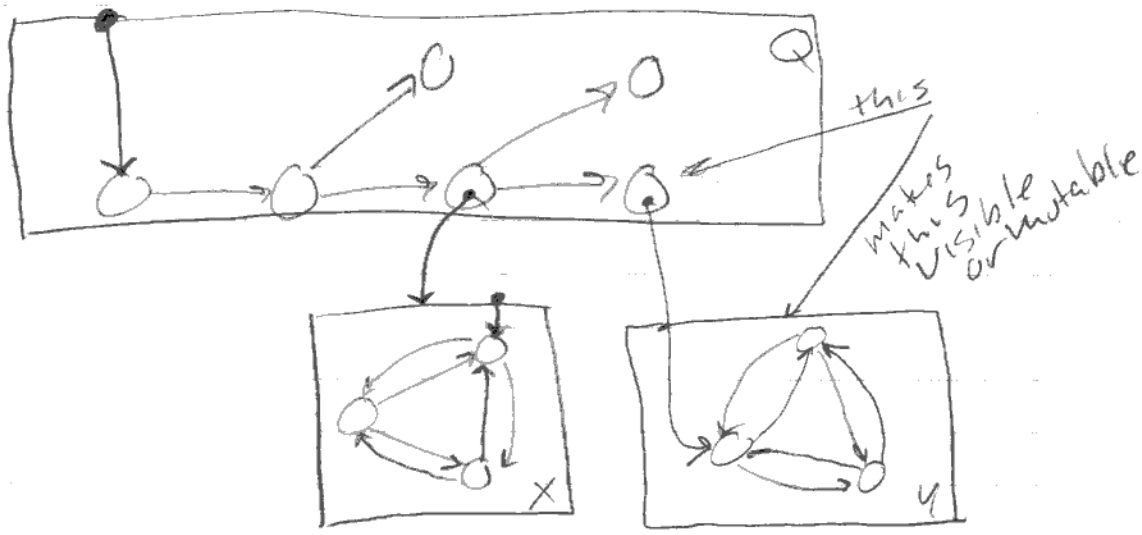


Turing machine cont



Turing

Fundamental Pattern



two independent but coupled data structures

Have data structure says what actions to take

Simplification — the substates in the cells are regular

the encapsulation is important

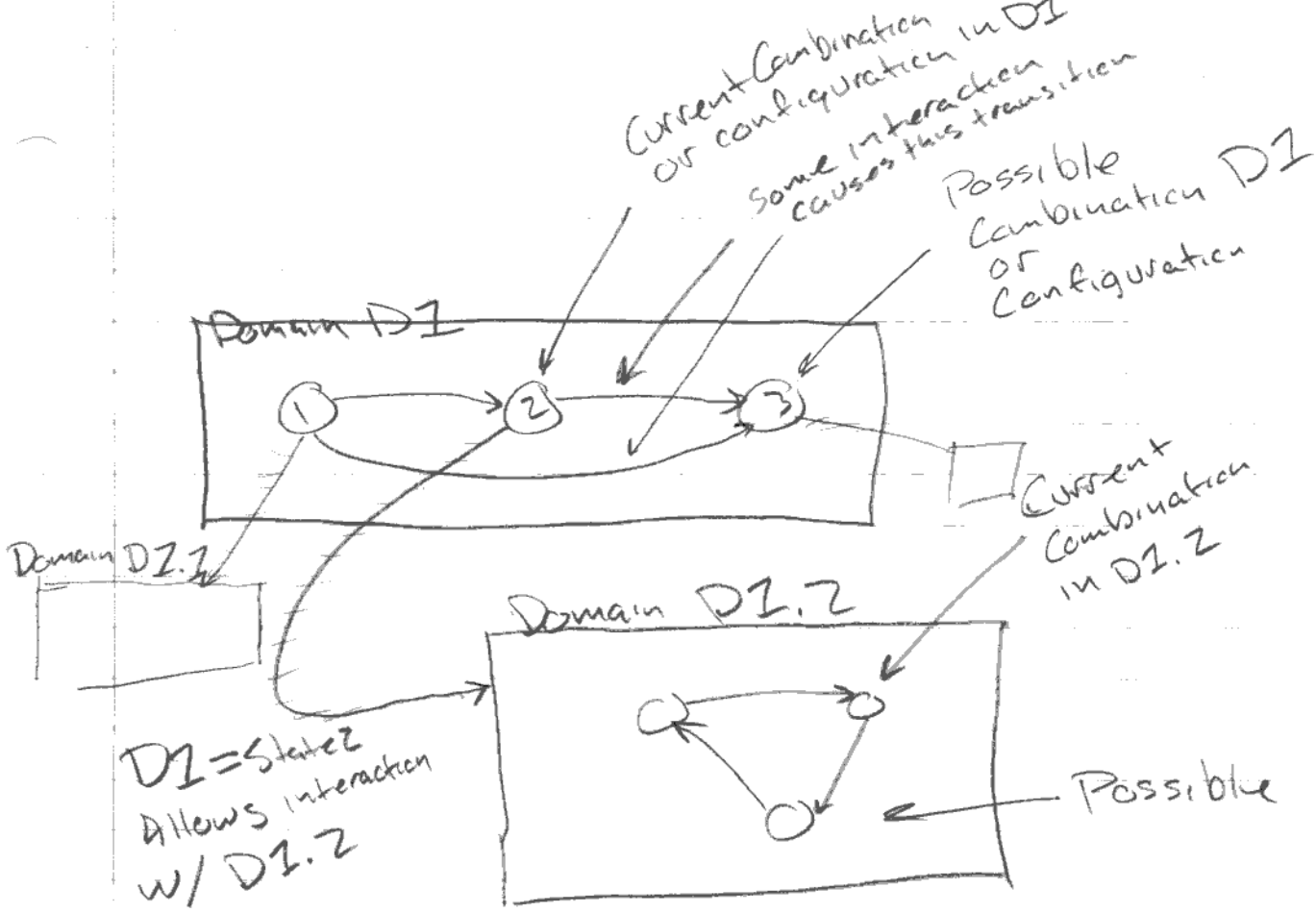
the nodes/states are a allowed combination of things

a edge is a move to a different allowed combination

The nodes in a graph are allowed combinations but only one node in the graph/combination is actualy established in a domain.

subgraphs can exist all the time but require super node to exist to be accessable

The nodes in a domain are mutually exclusive. --- only one at a time



Domain D1 has 1 active state or combination
D1 has many possible state or combo

A particle or something makes D1 unique such that D1 can only be in one state

In interesting situations (memory, computation)

Domain D1.2 exists while D1 in state 1 or 2 however active state in D1.2 can only have external impact when D1 == State 2 or be changed when D1 == state 2

A turing machine, and a machine in general

Algorithmically activates states within Multiple Domains

The program / machine will do the equivalent of a full search or enumeration of states in some domains

The programing is such that Domain A and B will be put into the prepper state at the same time so Domain C can transition to a desired state

In this way the computational machine narrows or drives the global states or global combinations of all Domains that are possible

Q) how do we choose the particles and the stable or allowed combinations that also have exclusivity in a domain