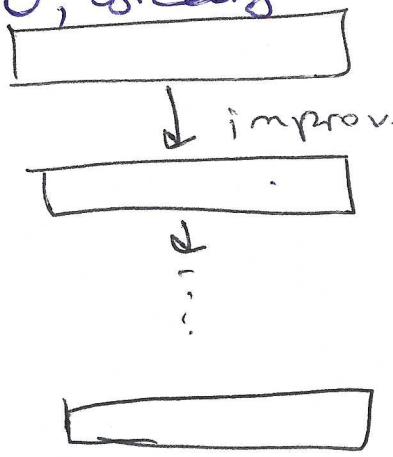
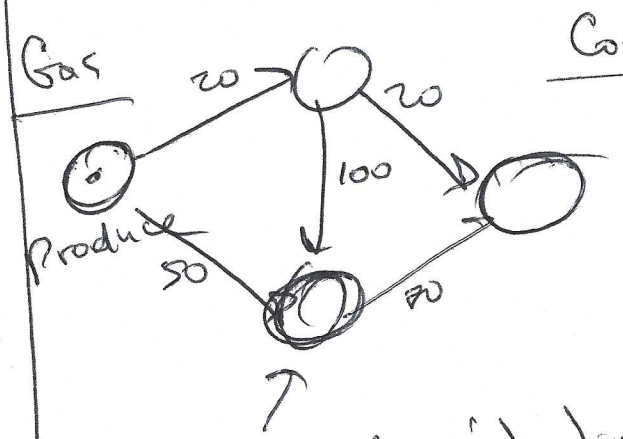
It. Impr

0, greedy ← how → cond. sol.

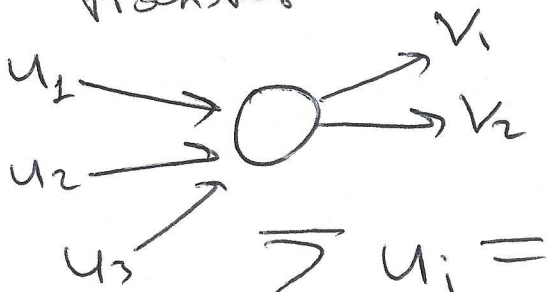


Max-Flow

Network DAG

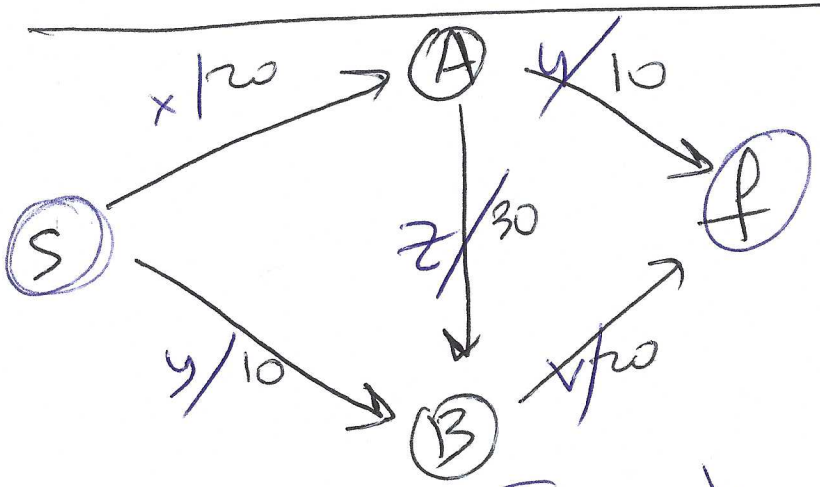


transfer station



$$\sum u_i = \sum v_j$$

constraint



$$\begin{cases} x = z + y \\ y + z = v \\ u, v, x, y, z \geq 0 \end{cases}$$

Target

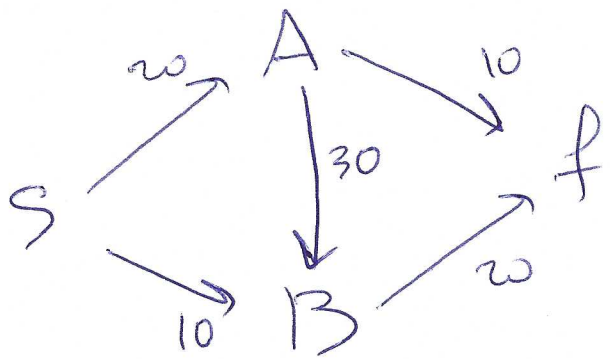
$x+y \to \max$

$u+v \to \max$

$x+y+z = z+u+v$

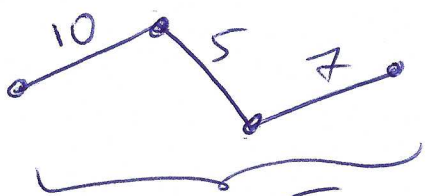
$x+y = u+v$

Ford-Fulkerson



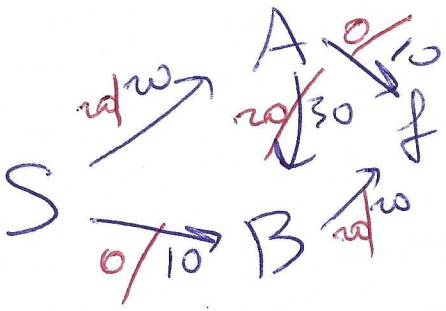
0) greedy = path max throughput

how →

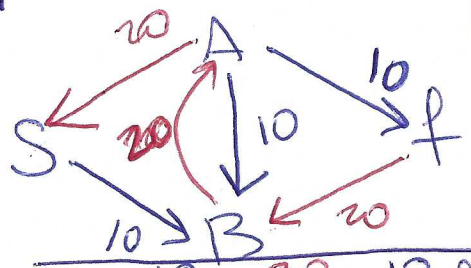


$S \xrightarrow{20} A \xrightarrow{30} B \xrightarrow{20} f$ BN=20 | Bottleneck

1) 20



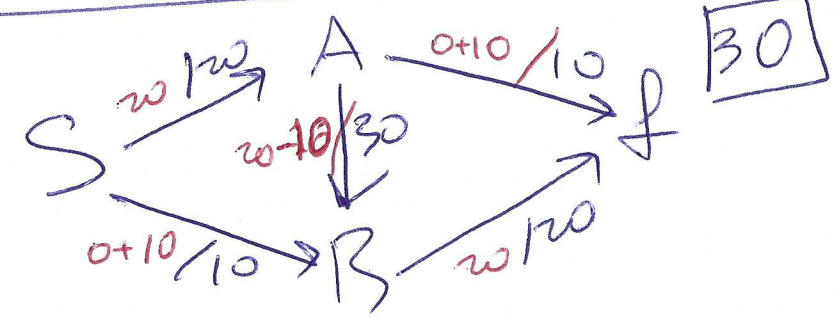
residual



find path in residual $S \xrightarrow{10} B \xrightarrow{20} A \xrightarrow{10} f$ | BN = 10

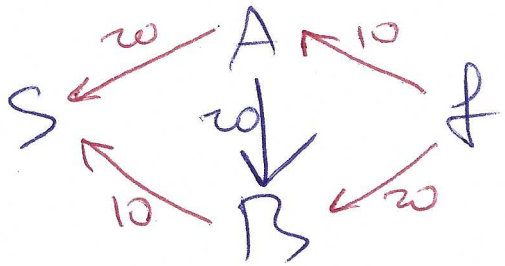
update flow using that path

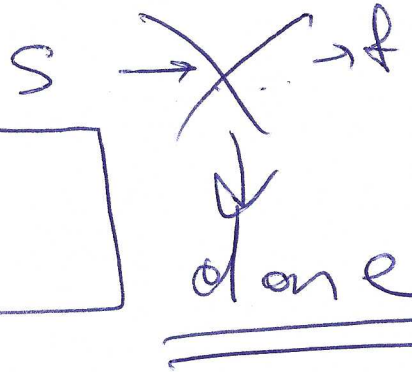
$S \xrightarrow{+10} B \xrightarrow{+10} A \xrightarrow{+10} f$



30

iter 2) residual

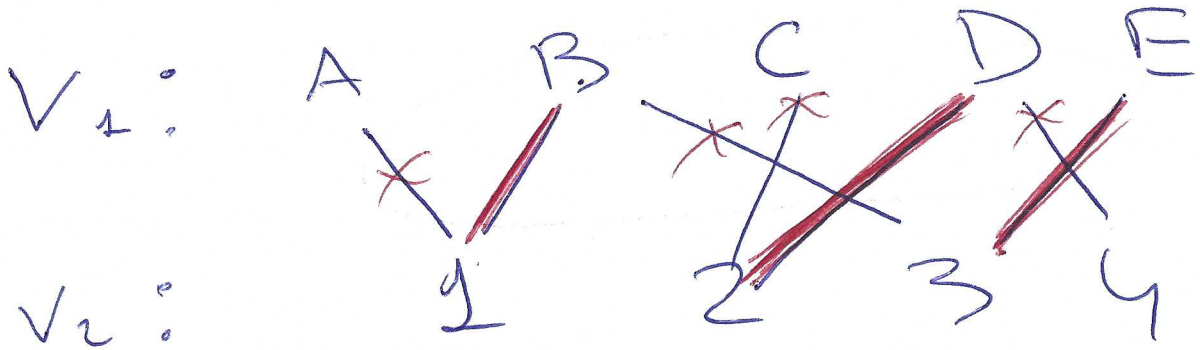




done

Bipartite matching (marriage)

Graph $V = V_1 \cup V_2$, each edge (v_1, v_2), $v_i \in V_i$



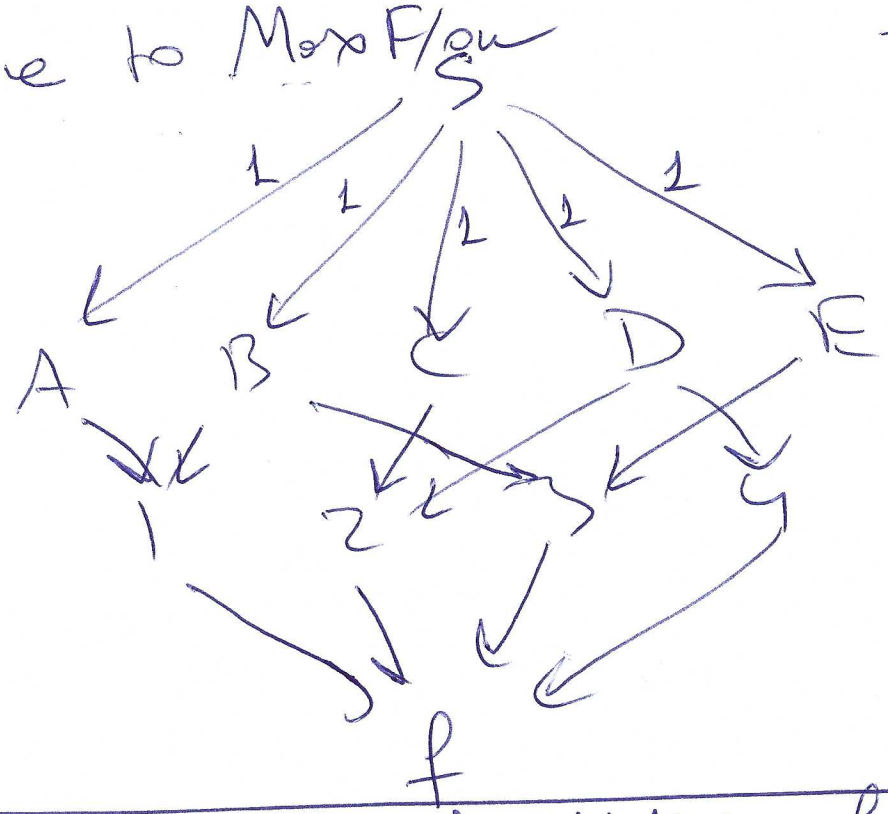
matching: correspondence V_1 to V_2 (find edges) that do not intersect (same vertex)

Ex. 1→B, B→3 do intersect

B-3, C-2 do not intersect

Max matching: largest subset of edges that form matching

Reduce to Max Flow

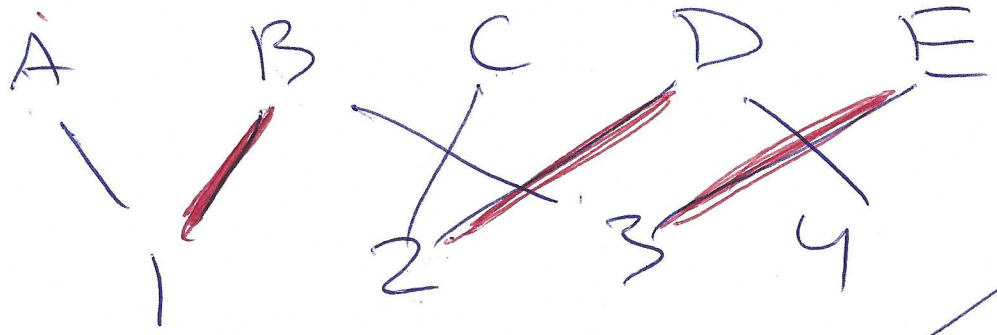


Max. Match
↓
Max-Flow
↓
LPP

Specialized Alg to MM: alternating path

0) greedy
1) improve ← no
2) done
↓

0)



$\#B = \#R + 1$ — 5 —

$\#B + \#R =$ odd

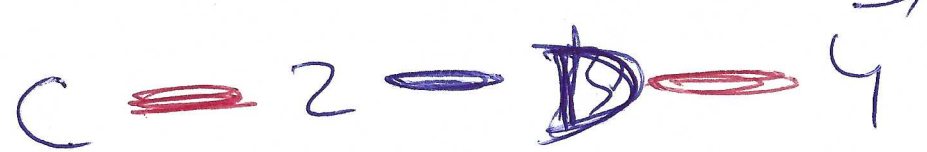
1)

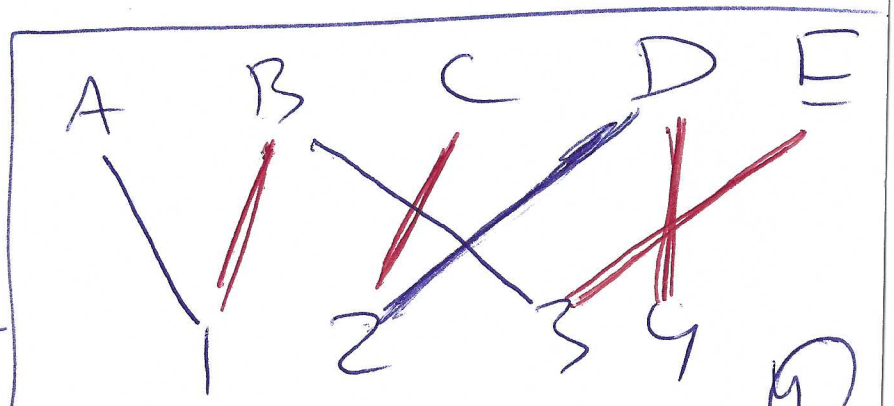
edge colors alternate

A — 1 — B — 3 — E ✗

C — 2 — D — 4 ✓ alternating!

Improve

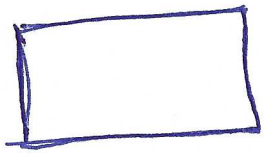


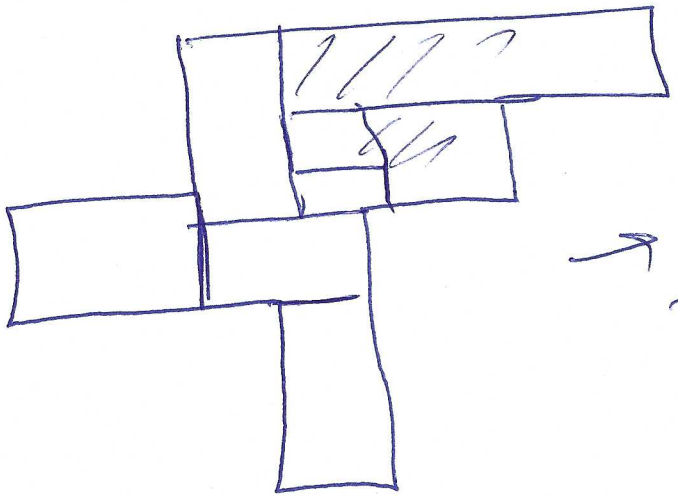


2) no more alt. path → done

(4)

axis-aligned polygon





→ min.

---

stable marriage problem — ordered list of preferred matches.

1 → 1) A, 2) B, 3) C

dynamic programming

Game Th, Nash Eq, Pareto Eq

Capitalism — Socialism

Mardin Tragedy of Commons.

Idea (not an actual class)

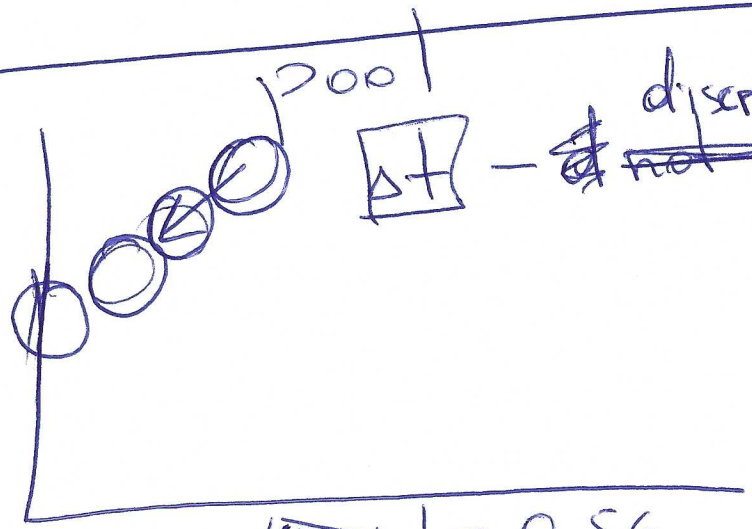
Event-driven programming design.

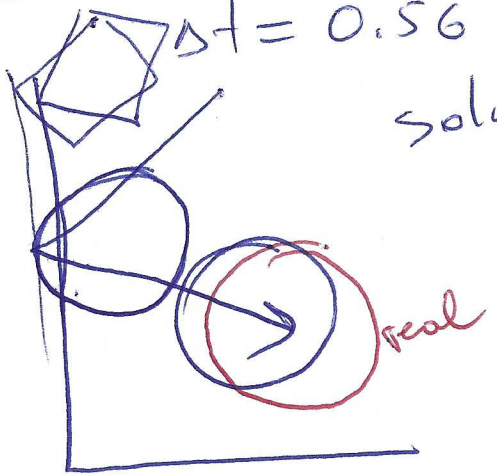
2) programming

math

1) alog

Dijkstra

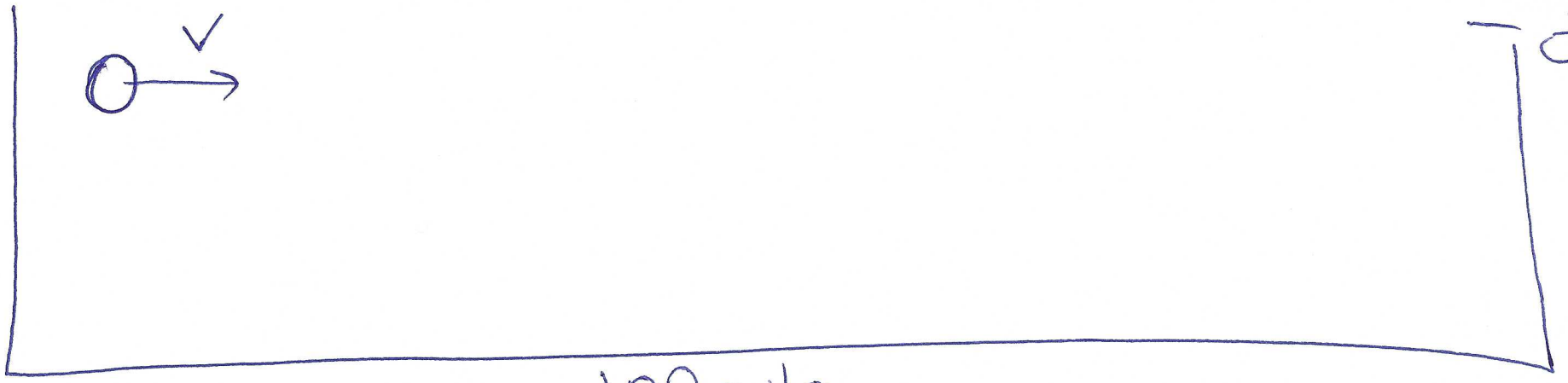




solutions 1) do nothing - no one will notice!!

2) go back in time } may be exact
Δt = -0.075 } messy + chain reaction

3) ED.



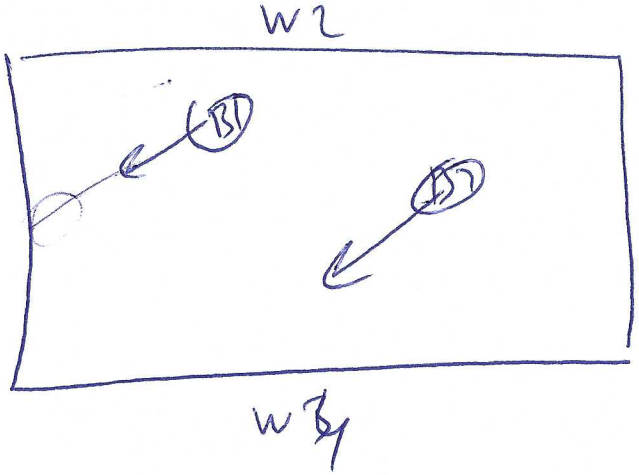
1) time to calc. pos in t=100 min.

a) Δt fixed $\quad \frac{100}{\Delta t} = \frac{100}{0.036} = \frac{100 \cdot 60}{1/60} = 360000$ frames

b) ED are there int. events during 100 min

yes → process then

no → final. pos = init. pos + 100×60×v

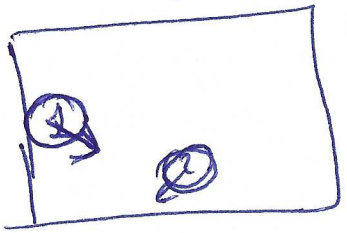


1) queue, priority of ~~even~~ future events

q = { (B1, W1, 0.1), (B2, W4, 0.2) }

e = (B1, W1, 0.1)

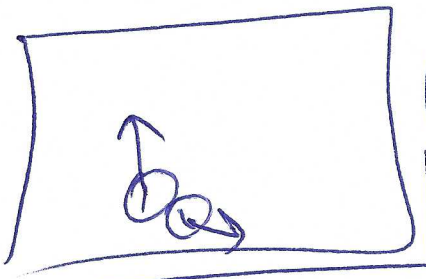
```
init pqe
while(!pqe.empty())
    e = pqe.pop();
    // process event
       recalc. position
               vectors.
    // new events ?
    pqe.insert( new_event);
}
```



B1, B2, 0.19

q = { (B1, B2, 0.19), (B2, W4, 0.2) }

e = (B1, B2, 0.19)



(B1, W2, ?)
(B2, W4, ?)

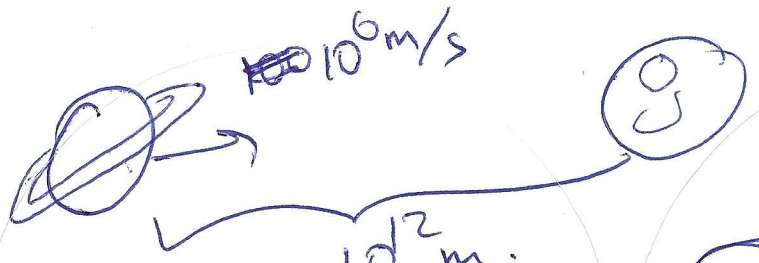
e = (B2, W4, 0.2) → cancelled !!

pool

1) planets

real time Billions of years

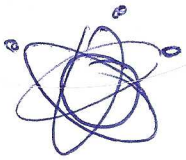
processing time 2 s.

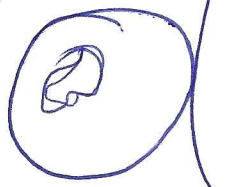


2) atoms

real t = 1 ms

process = months !!!





BiTe

Scientific

Simulation