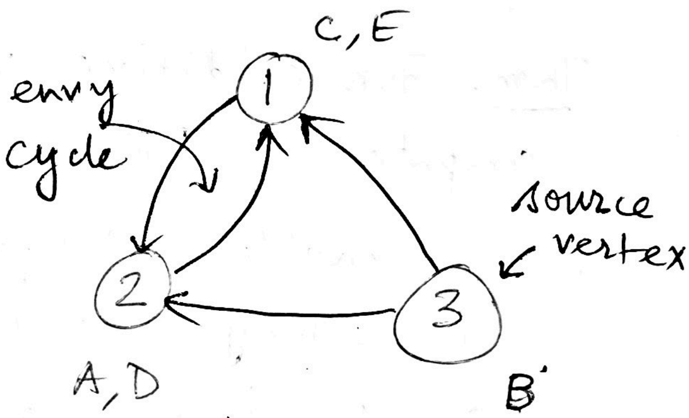
Envy-cycle elimination (for monotone ~~add~~ valuations)

Envy-graph of an allocation

- vertices = agents
- edges = from i to j if i envies the bundle of agent j in that allocation.

Our running example will be using additive valuations for simplicity but this is not necessary for the construction of an envy-graph. The designer can just query the agent with the bundle and get the valuation.

	A	B	C	D	E
1	4	2	5	4	1
2	1	0	5	1	1
3	1	1	5	1	1



Envy-graph

Source vertex: doesn't have any incoming envy edge.

Envy-cycle elimination algorithm

While ~~is~~ there is an unallocated object:

if the envy graph has a source vertex, assign the object to that agent

else resolve envy cycles until a source vertex shows up and assign the good to her.

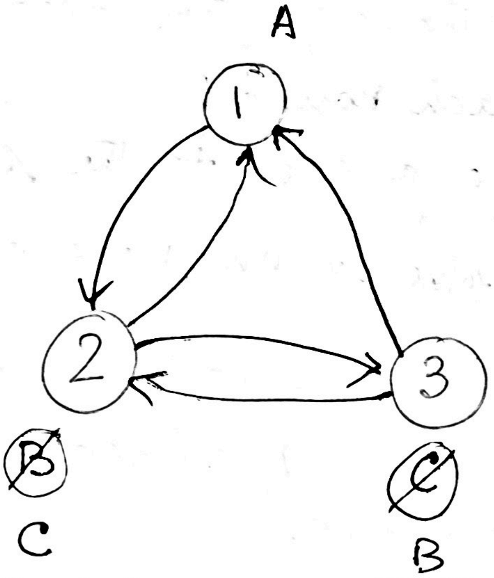
~~Repeat~~

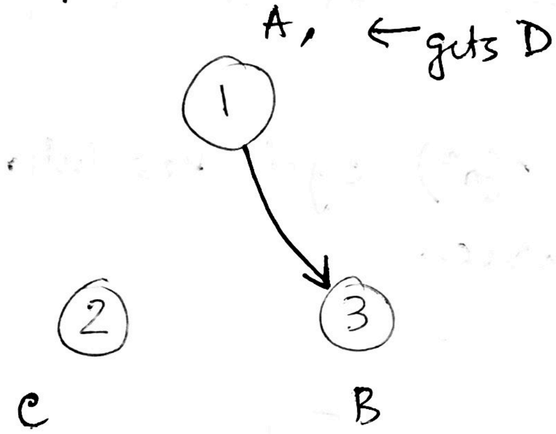
~~Resolve envy cycles~~.

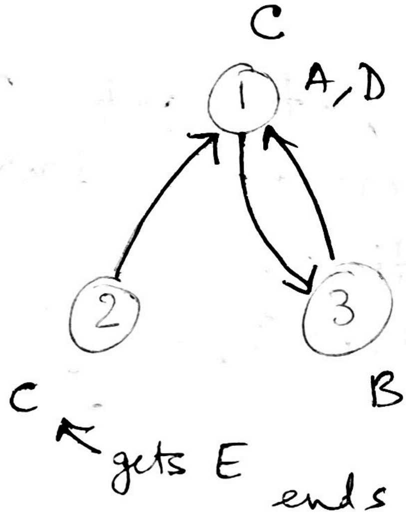
Resolve a cycle: Give the bundle to the agent that is pointing to it, i.e., move bundles in the reverse direction.

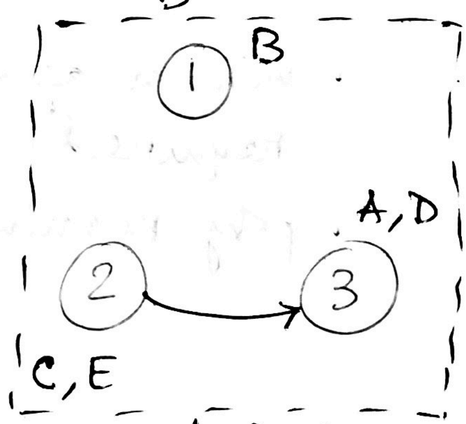
Example:

	A	B	C	D	E
1	0	2	0	1	1
2	1	2	5	10	1
3	1	4	2	10	1









~~ends~~ Pareto better than that

Final allocation ~~({B}, {C, E}, {A, D})~~

Check if this is EF1? ({A, D}, {C, E}, {B})

drop D from {A, D} → agent 2 does not envy any more.

Questions: Does the algorithm terminate?

In poly time?

Is the algorithm correct? i.e., gives an EF1 allocation?

9-3

Q1: Does the algorithm terminate?

In each round:

- give a good to the source vertex — at most m goods.
- Resolve an envy ~~cle~~ cycle —

Thm: (To show) After resolving any envy cycle, the number of edges in the envy graph ~~so~~ strictly decreases.

- with n agents, at most $O(n^2)$ cycle resolutions required to create a source.
- poly running time.

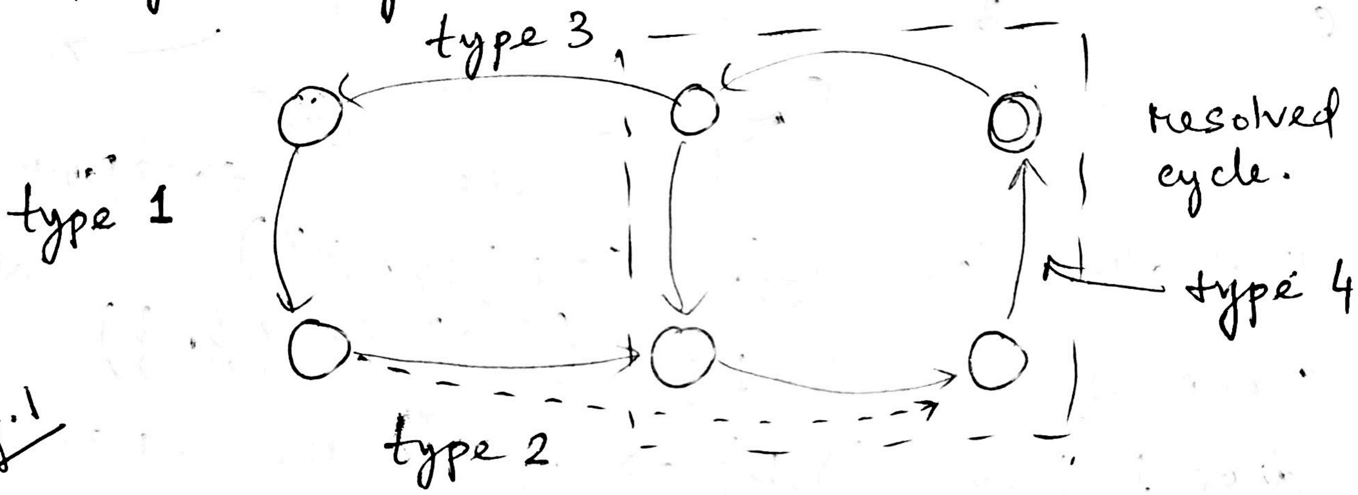
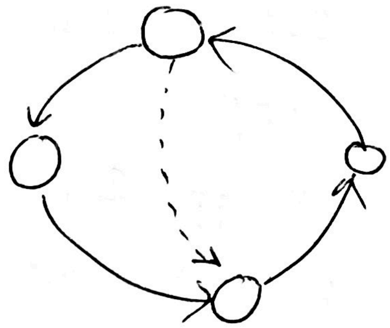


Fig. 1

- type 1 edges are unaffected. — # unchanged
- type 2 edges head move backwards in the resolved cycle.
  # unchanged
- type 3 edge may disappear or may stay — note (edges are from agents to envy bundles — so root can't move. After the cycle resolution, the root agent may be happier and stops envying, but does not have to. In the worst case # ~~rem~~ unchanged.

- type 4 edges: The original cycle edges ~~are gone~~ disappear. The dashed edges may stay or disappear, but such edges can't increase. Hence at least one edge will disappear from the original envy graph.



Summary: ECE terminates in poly time.

Q2: Is the algorithm EF1?

$\forall i,j, \exists x_j \in A_j$ s.t.

$$v_i(A_i) \geq v_i(A_j \setminus \{x_j\})$$

Argue that in each iteration "preserves" EF1

While there is an unallocated object

① • if the envy graph has a source vertex, assign the object to that agent.

② • else resolve envy cycles until a source vertex shows up and assign the object to that agent.

— Step 1: If we assign an object to a source — ~~the~~ who wasn't envied by ~~anyon~~ anyone, — it can be EF1 but can't violate EF1

- Step 2: For type 1 edges, nothing changes, hence EF1.

For type 2 edges — The bundles are shifted around, and hence not changing the EF1 in the agents that were pointing to agents in the cycle.

For type 4 edges — ~~they~~ agents are strictly happier, they don't EF1 envy any other agent.

- Agents who are outside the cycle, their EF1 relation remains identical. Bundles are never broken, ~~hence~~ they are only shifted ownership.
- Agents who are inside the cycle, they are getting happier, hence either their EF1 improves i.e., the magnitude of envy reduces or envy disappears. In either case, the EF1 condition holds.

Remarks:

- We never used additivity.
- Only used monotonicity

$$v_i(S) \leq v_i(T), \ \forall \ S \subseteq T.$$

more items are weakly preferred.

Complexity is in terms of query.

Fairness requires some more conditions – even an empty allocation is EF.

Minimum requirement should be completeness.

– Assign all items to at least one agent.

But completeness alone is not sufficient

	A	B	C	D	E
1	$\boxed{4}$	3	$\boxed{1}$	1	$\boxed{1}$
2	5	$\boxed{2}$	1	$\boxed{1}$	1

Fig. 2

☐ is improved via ⬚ Pareto improved.

Pareto Optimality: An ~~A~~ allocation A is PO if

$\nexists B$ s.t. $v_i(B_i) \geq v_i(A_i)$ $\forall i \in N$

and $\exists j$ s.t. $v_j(B_j) > v_j(A_j)$.

If we want to improve the allocation for some agent it has to come at an expense of someone else.

note that PO alone is not meaningful either. The grand bundle allocated to one agent is PO as well.

We need both FAIR and EFFICIENT allocation.

Is EF1 and PO achievable together?

Attempt 1: Round-Robin: example Fig. 2 above

Attempt 2: Envy-cycle elimination:

	A	B	C
1	4	3	1
2	5	2	1

$\xrightarrow{\text{Pareto improve}}$

	A	B	C
1	4	$\boxed{3}$	$\boxed{1}$
2	$\boxed{5}$	2	1

~~Attempt 3~~: