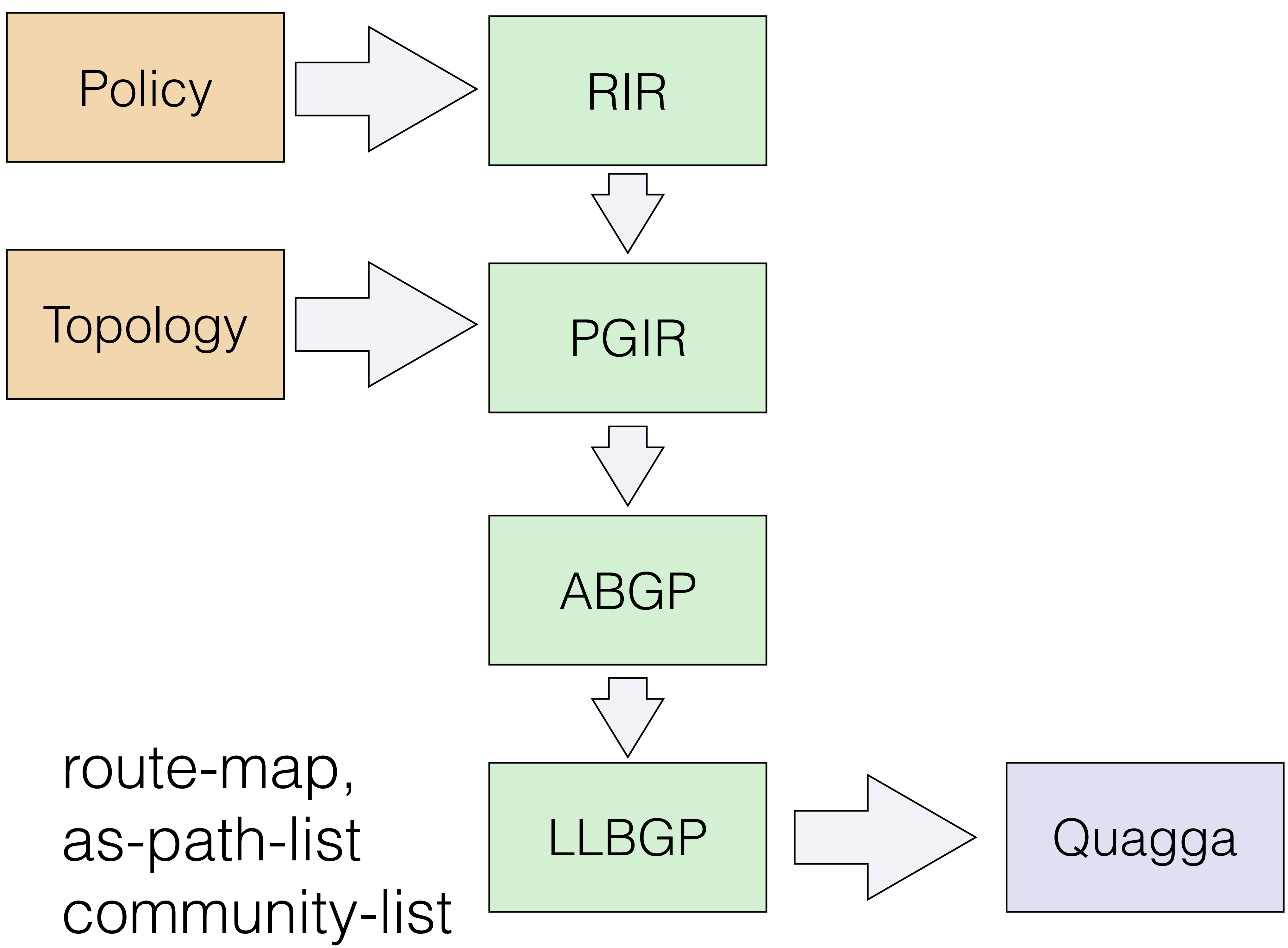
Implementation



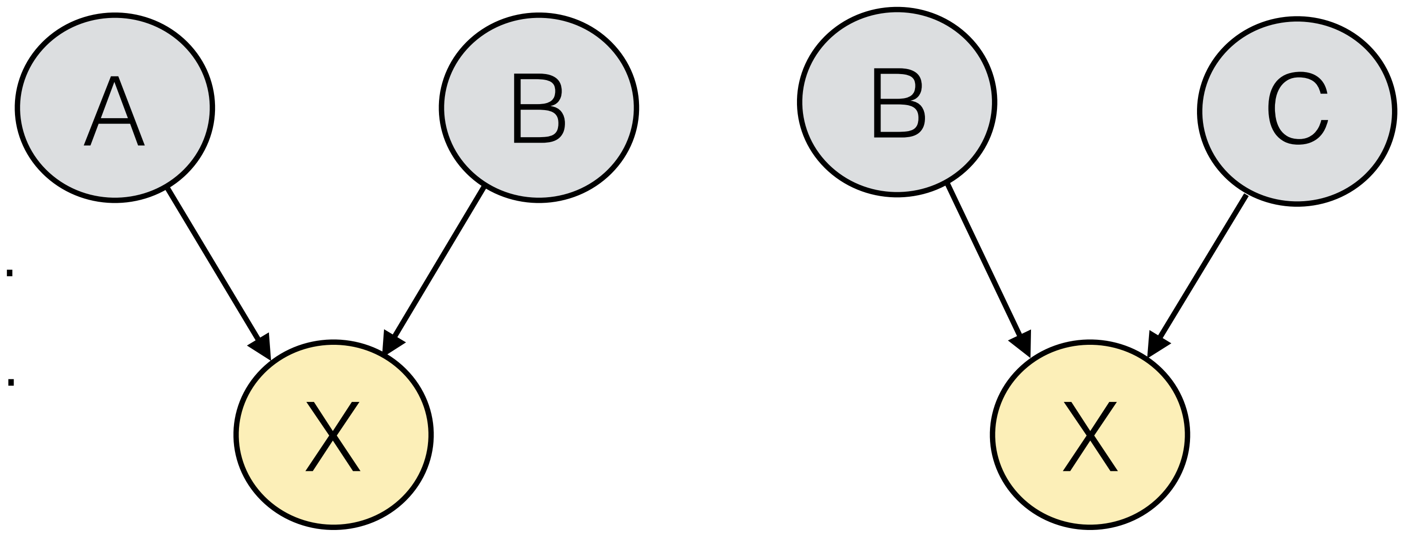
Quagga

```
ip prefix-list pl-1 permit 0.0.0.0/0 ge 0 le 32
!
ip community-list standard cl-1 permit 100:1
ip community-list standard cl-2 permit 100:2
!
ip as-path access-list path-1 permit (^103_ | ^102_)
!
route-map rm-in permit 10
  match community cl-1
  match ip address prefix-list pl-1
  set community additive 200:14
!
route-map rm-in permit 20
  match community cl-2
  match ip address prefix-list pl-1
  set local-preference 99
  set community additive 200:14
!
```

Quagga

match comm=1, peer=A,B, …

match comm=2, peer=B,C, …



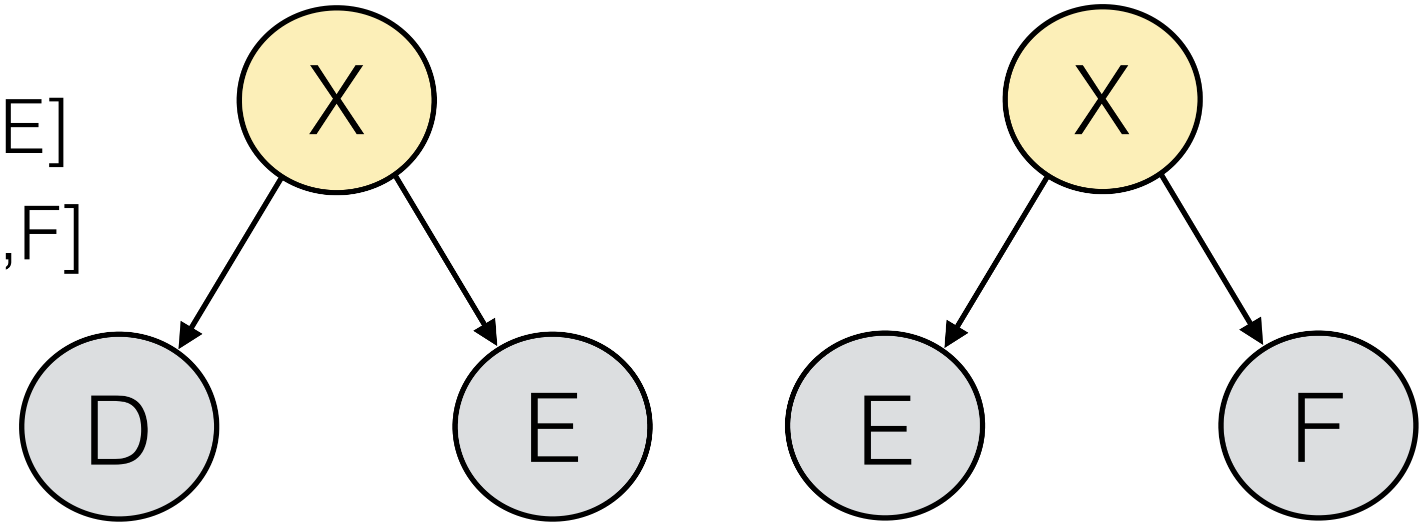

```
ip as-path access-list peerAB permit (^A_ | ^B_)
ip as-path access-list peerBC permit (^B_ | ^C_)
ip community-list standard cl-1 permit 100:1
ip community-list standard cl-2 permit 100:2

route-map rm-in permit 10
  match as-path peerAB
  match community cl-1
  set community additive 100:3
!
…
```

Quagga

match comm=1, peer=A,B [export D,E]

match comm=2, peer=B,C, [export E,F]



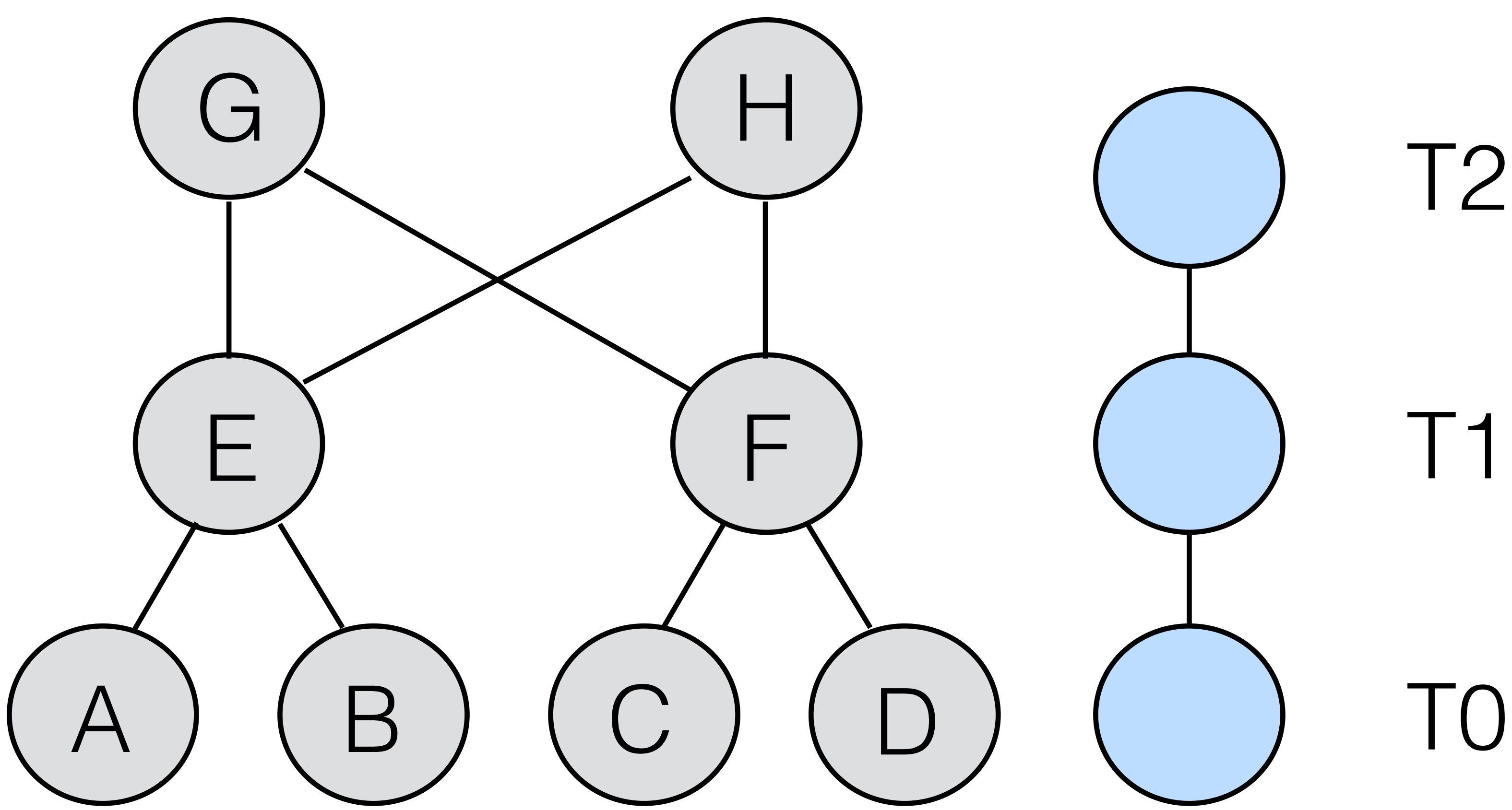
How do we know when to export to a peer?
Need a per-peer export policy

(1) New route-map for every peer
(2) Assign community for at import filter for each peer.

Abstract Topology

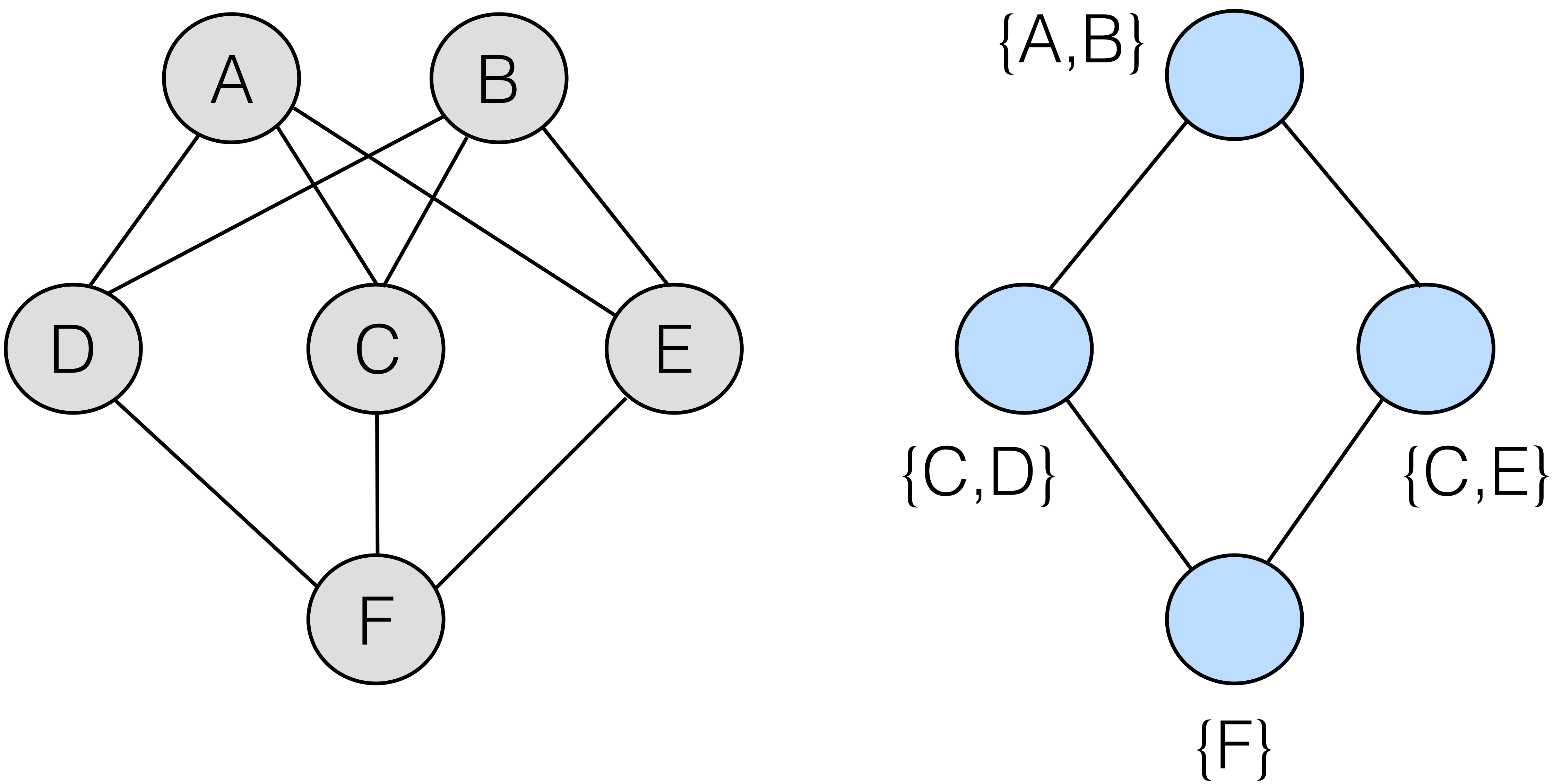
Idea:
Each abstract node corresponds to a set of concrete nodes

Example: T2; T1; T0 (G + H); (E + F); (A + B + C + D)



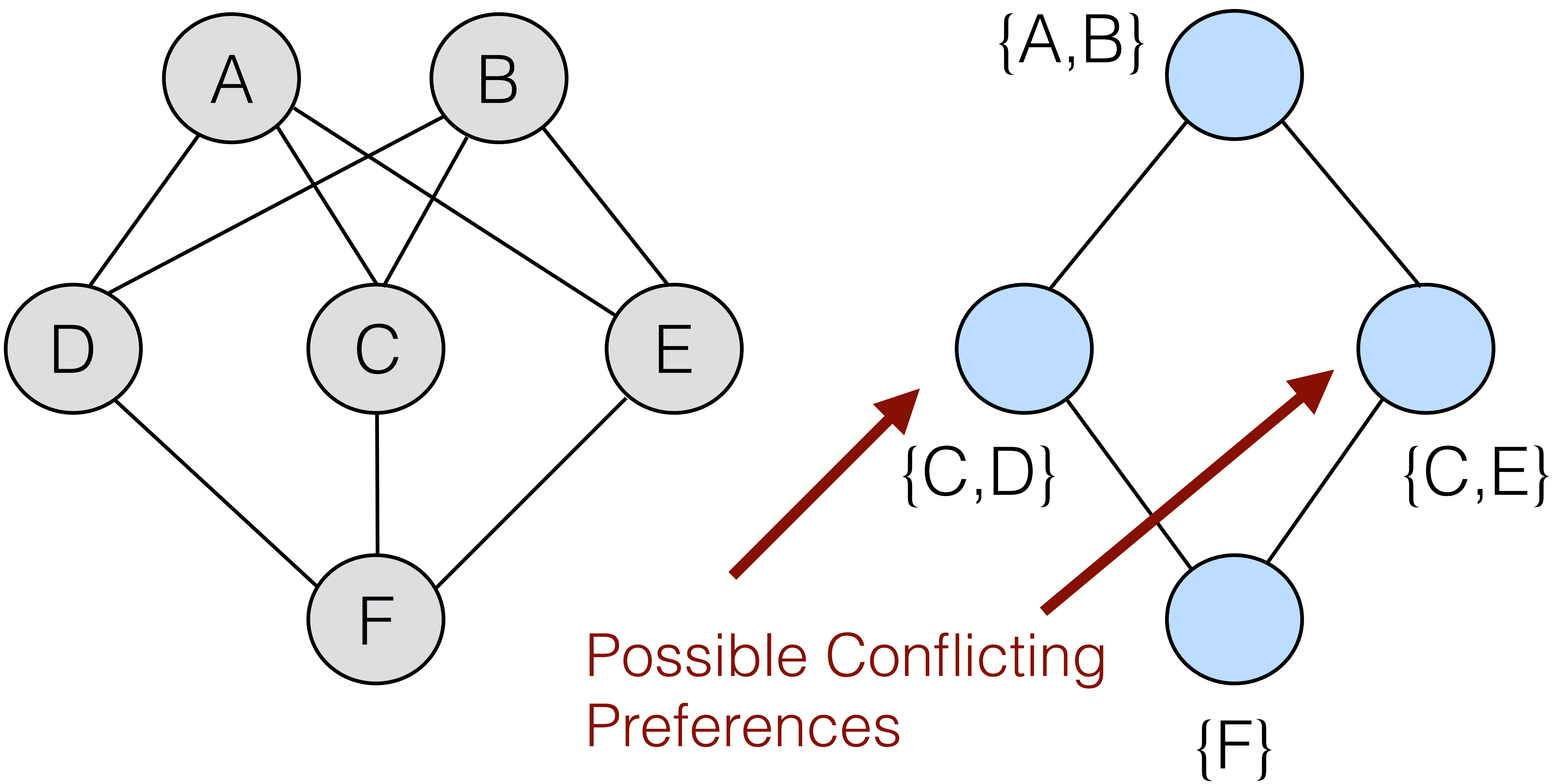
Abstract Topology

Attempt 1:
Each abstract node corresponds to a set of concrete node



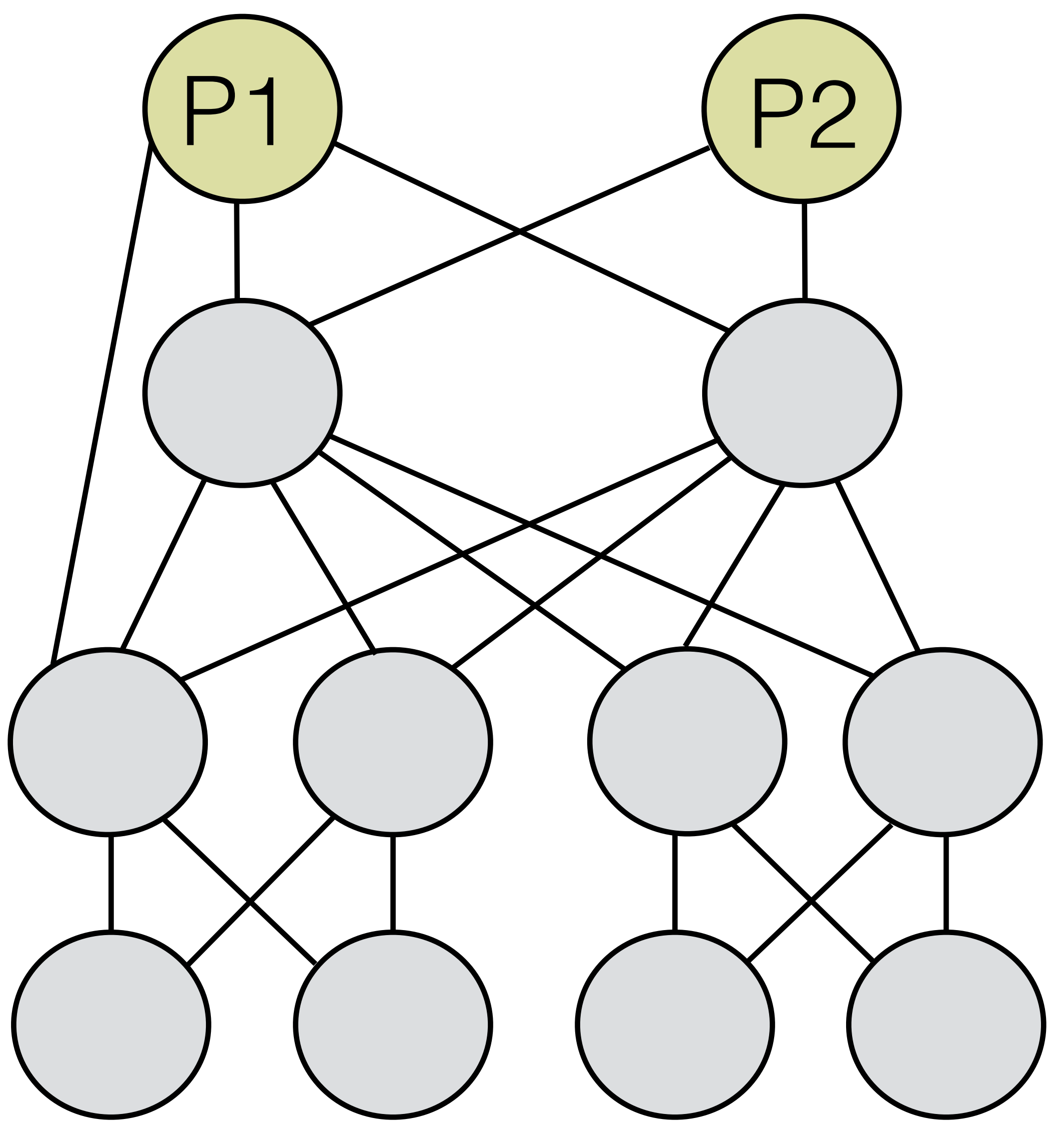
Abstract Topology

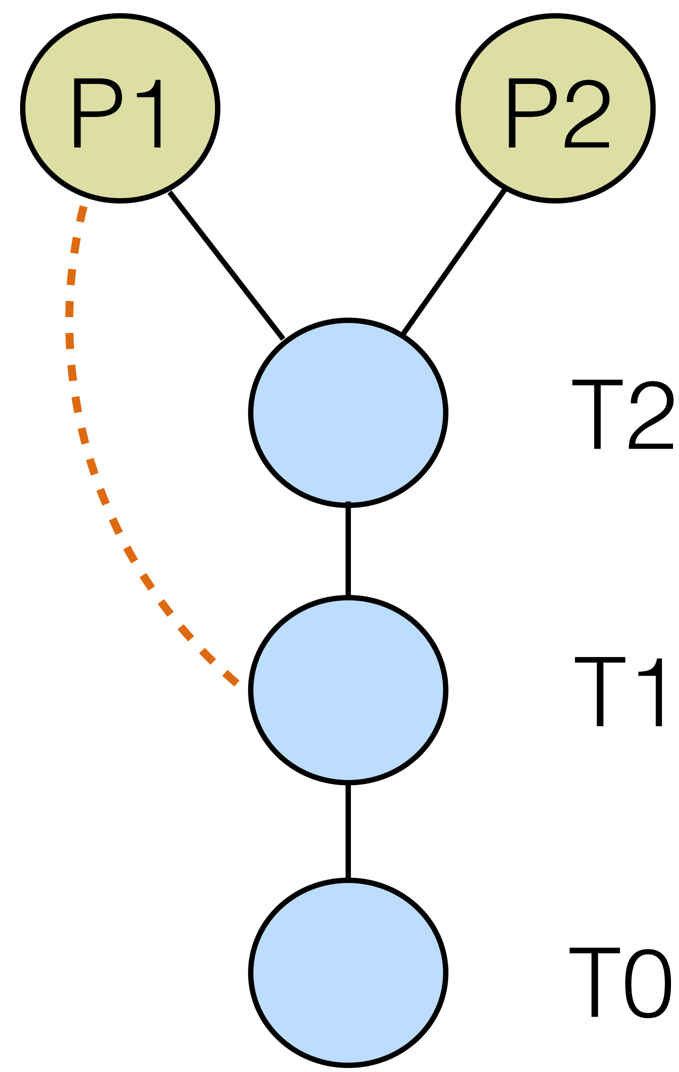
Attempt 1:
Each abstract node corresponds to a set of concrete node



Abstract Topology

Attempt 2:
Not including a link in the abstract topology

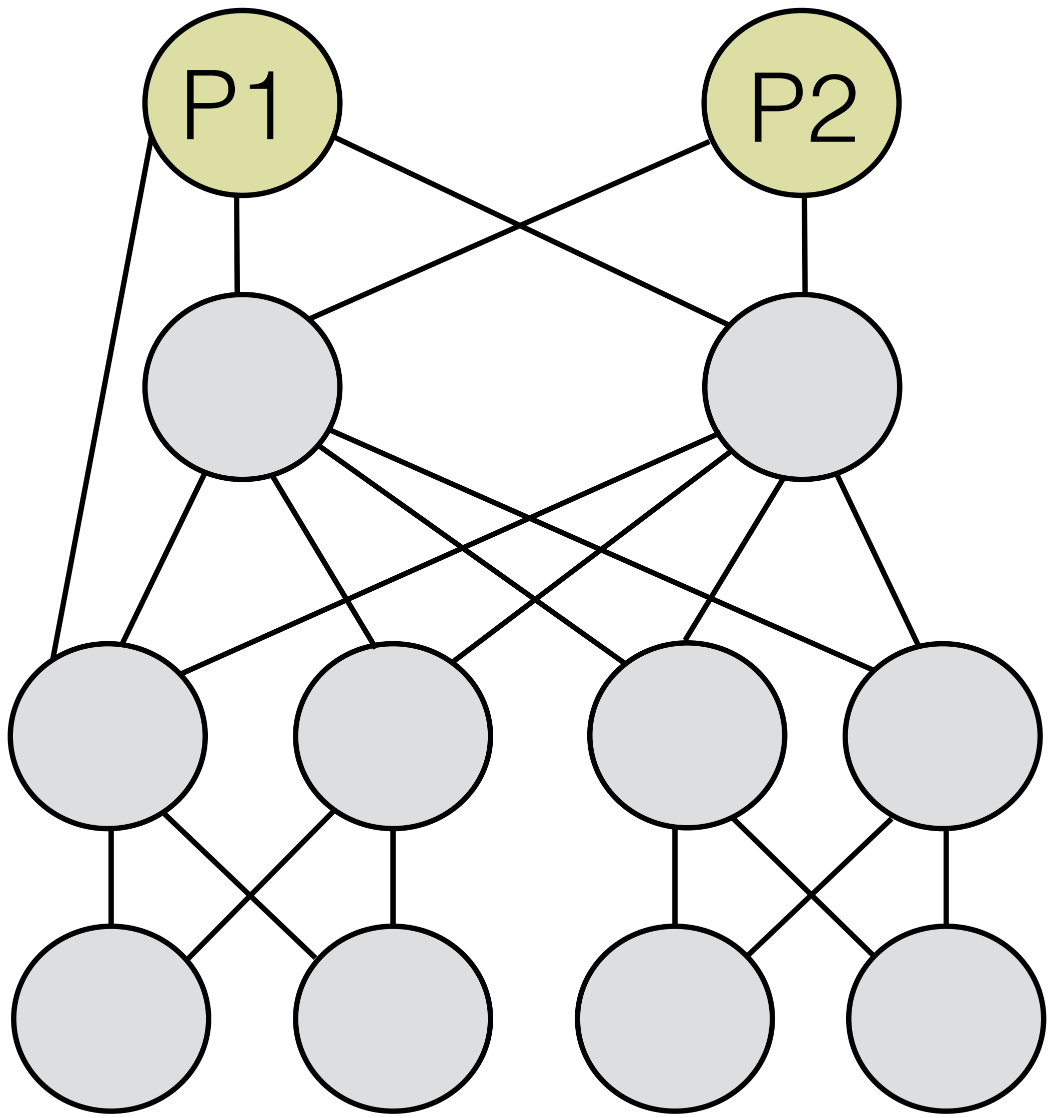


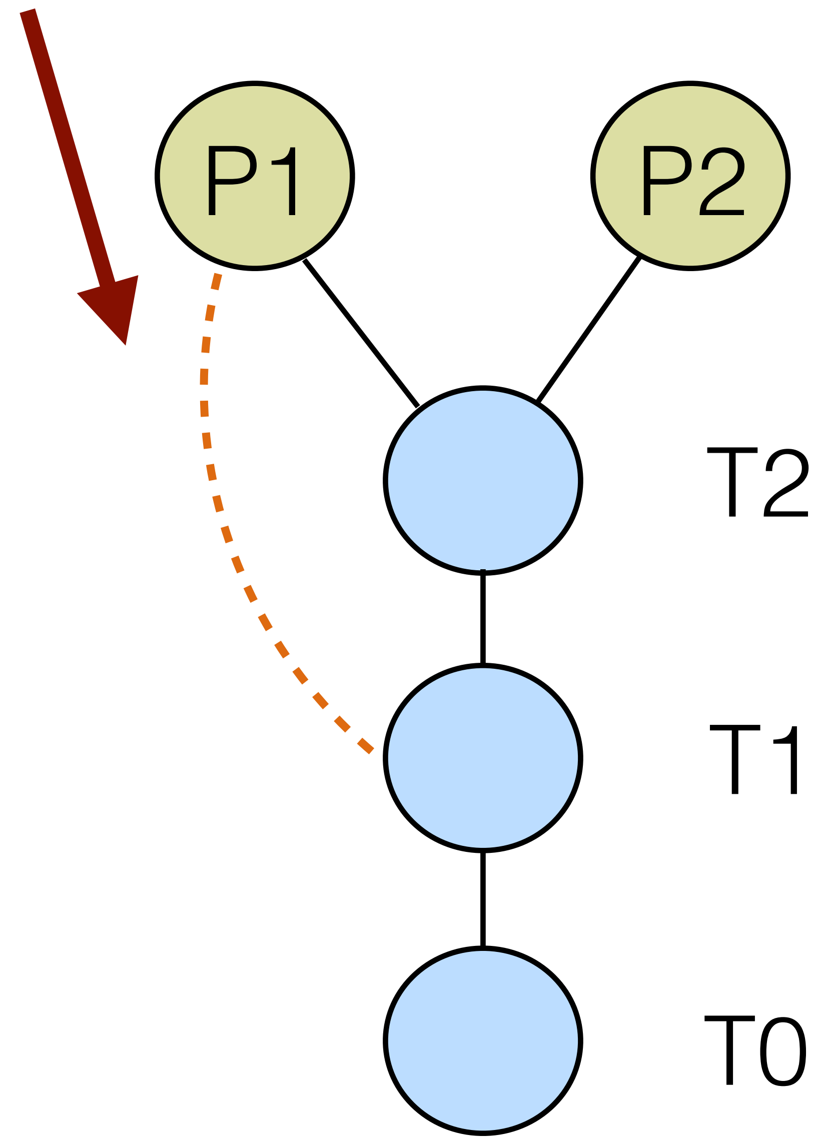


Abstract Topology

Policy: out; T1; T0

Path exists in concrete network but won't be allowed in abstract network





Abstract Topology

Criteria:

Concrete Topology Tc: (Vc, Ec)
Abstract Topology Ta: (Va, Ea)

Looking for a mapping f: Vc —> Va
s.t. if v1,…,vn in Tc then f(v1),…,f(vn) in Ta

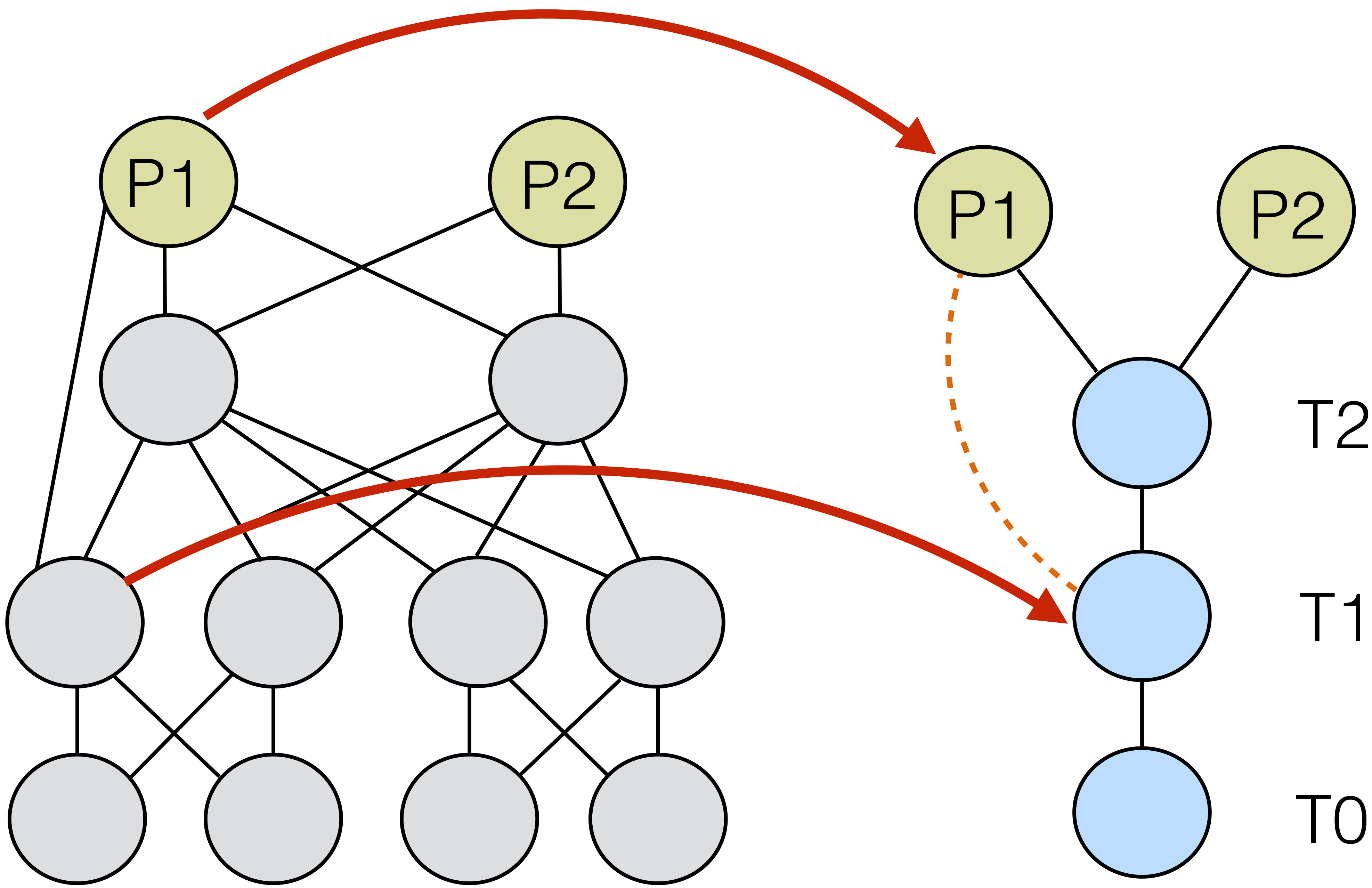
Graph Homomorphism

mapping f: Vc —> Va
s.t. if (x,y) in Ec then f(x),f(y) in Ea

Satisfies the above criteria

Abstract Topology

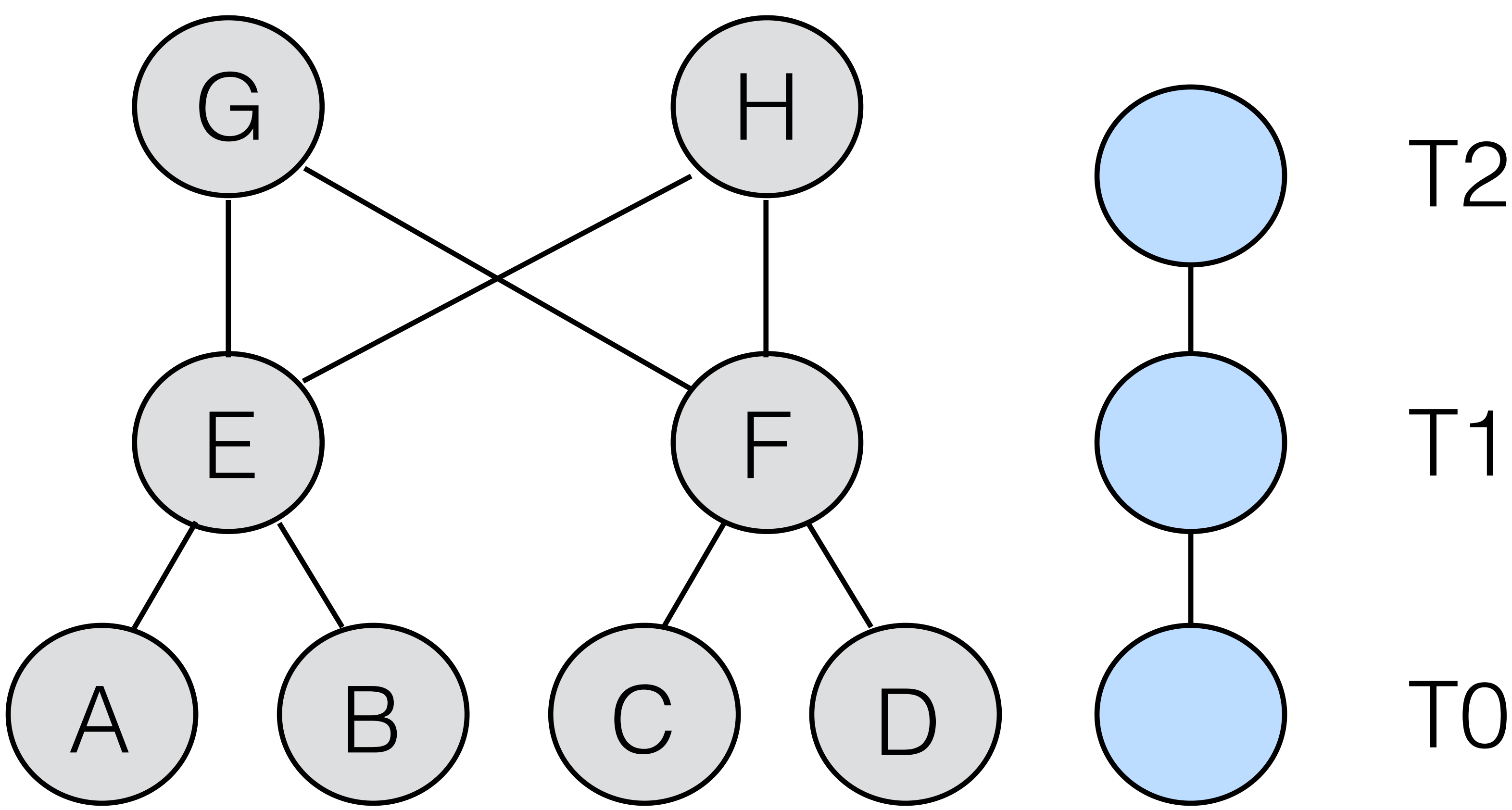
Policy: out; T1; T0



Abstract Topology

Correctness:
Each abstract node corresponds to a set of concrete nodes

Example: T2; T1; T0 (G + H); (E + F); (A + B + C + D)



Correctness

Correctness:
Concrete alphabet, Σc
Abstract alphabet Σa
Abstract policy p

$\text{compile}(p[x_i \,/ \cup f^{-1}(x_i)])$

$$\begin{tikzcd}[column sep=huge, row sep=huge]
p \arrow[r, "\text{compile}"] \arrow[d, "\substack{\text{substitute}\\ \text{abstract for}\\ \text{concrete}}"'] & {} \arrow[d, "\substack{\text{substitute}\\ \text{abstract for}\\ \text{concrete}}"] \\
{} \arrow[r, "\text{compile}"'] & {}
\end{tikzcd}$$

Correctness

Correctness:

Concrete alphabet, Σc

Abstract alphabet Σa

Abstract policy p

$$\text{compile}(p[x_i \;/\; \cup\; f^{-1}(x_i)]) \;=\; \text{compile}(p)[x_i \;/\; \cup\; f^{-1}(x_i)])$$

Policy substitution

ABGP substitution

Correctness

Observations:

For every path through Tc, there is path through Ta
Every path through PGc is a valid path through Tc
Every path through PGa is a valid path through Ta
For every path through PGc, there is a path through PGa

Idea:

Show that whenever certain properties hold in the abstract PG, they also hold in the concrete PG

Correctness

Observations:

For every path through Tc, there is path through Ta
Every path through PGc is a valid path through Tc
Every path through PGa is a valid path through Ta
For every path through PGc, there is a path through PGa

Idea:

Show that whenever certain properties hold in the abstract PG, they also hold in the concrete PG

Correctness

Observations:

For every path through Tc, there is path through Ta
Every path through PGc is a valid path through Tc
Every path through PGa is a valid path through Ta
For every path through PGc, there is a path through PGa

Idea:

Show that whenever certain properties hold in the abstract PG, they also hold in the concrete PG

Correctness

Property X:

Given a PG that compiles and nodes X1, X2 such that shadows(X1,X2) and X1 ≥ X2
for every path $p = a_1,\ldots,a_i$, X1, $a_{i+1},\ldots,a_n$ either
(1) there exists a path $q = b_1,\ldots,b_j$, X2, $a_{i+1},\ldots,a_n$ where $b_1,\ldots,b_j$, X is simple
(2) there is no simple path to X2
(3) there exists a path $q = b_1,\ldots,b_{j-k}, a_{i+1+h},\ldots,a_n$
with rank(p) ≥ rank(q)

Property X ⟶ Completeness

Property X (PGa) ⟶ Property X (PGc)