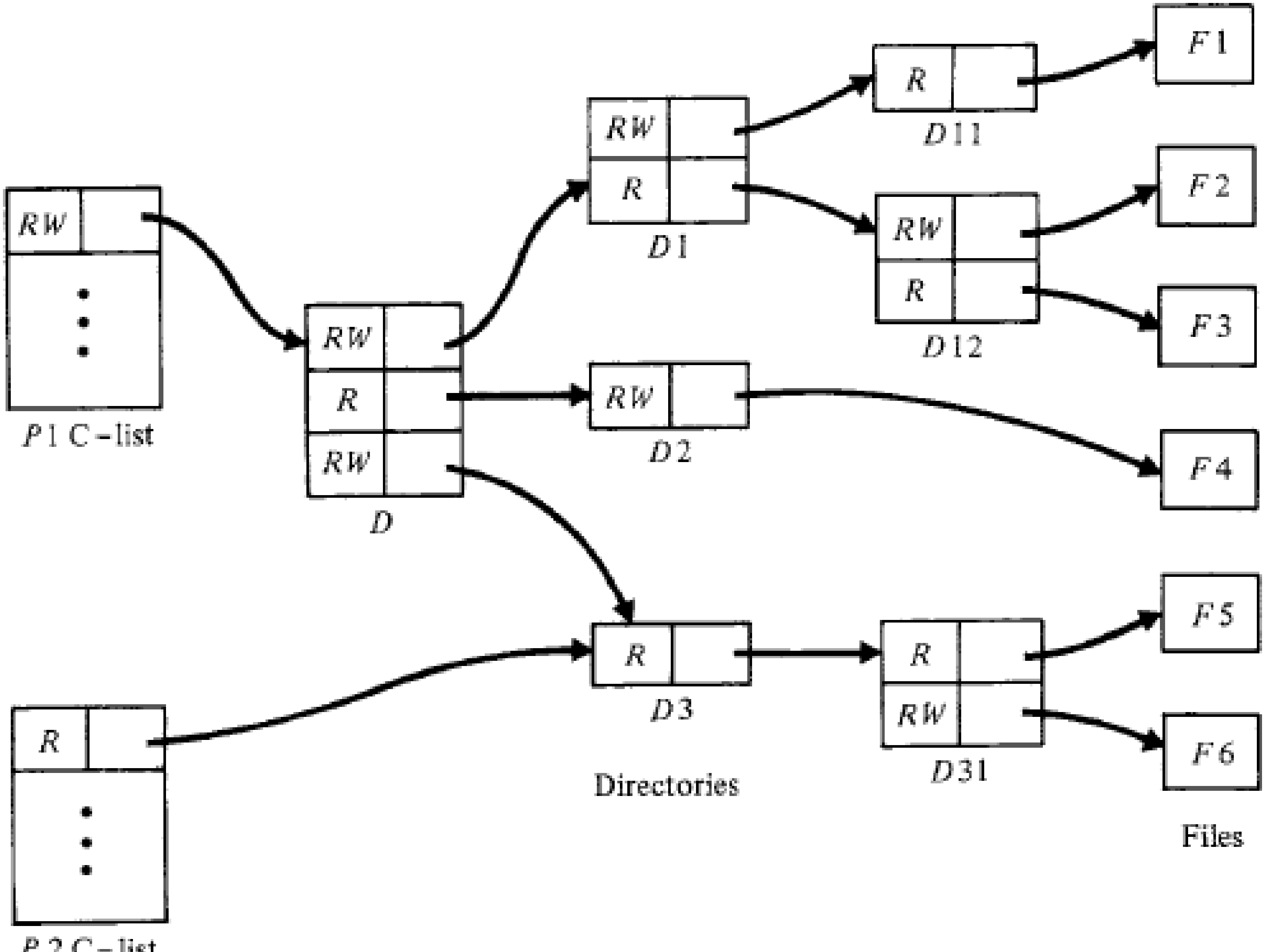
Take Grant System

Take Grant Graph Model

Jones et al – Capability system (Hydra)

- Nodes of the Graph : Subjects and objects
  - subjects are not objects in this model
- Edges: rights
  - two special rights: take (t), and grant (g).
- If a subject s has “t” for an object x, then it can take any of x's rights;
- if it has “g” for x, then it can share any of its rights with x.

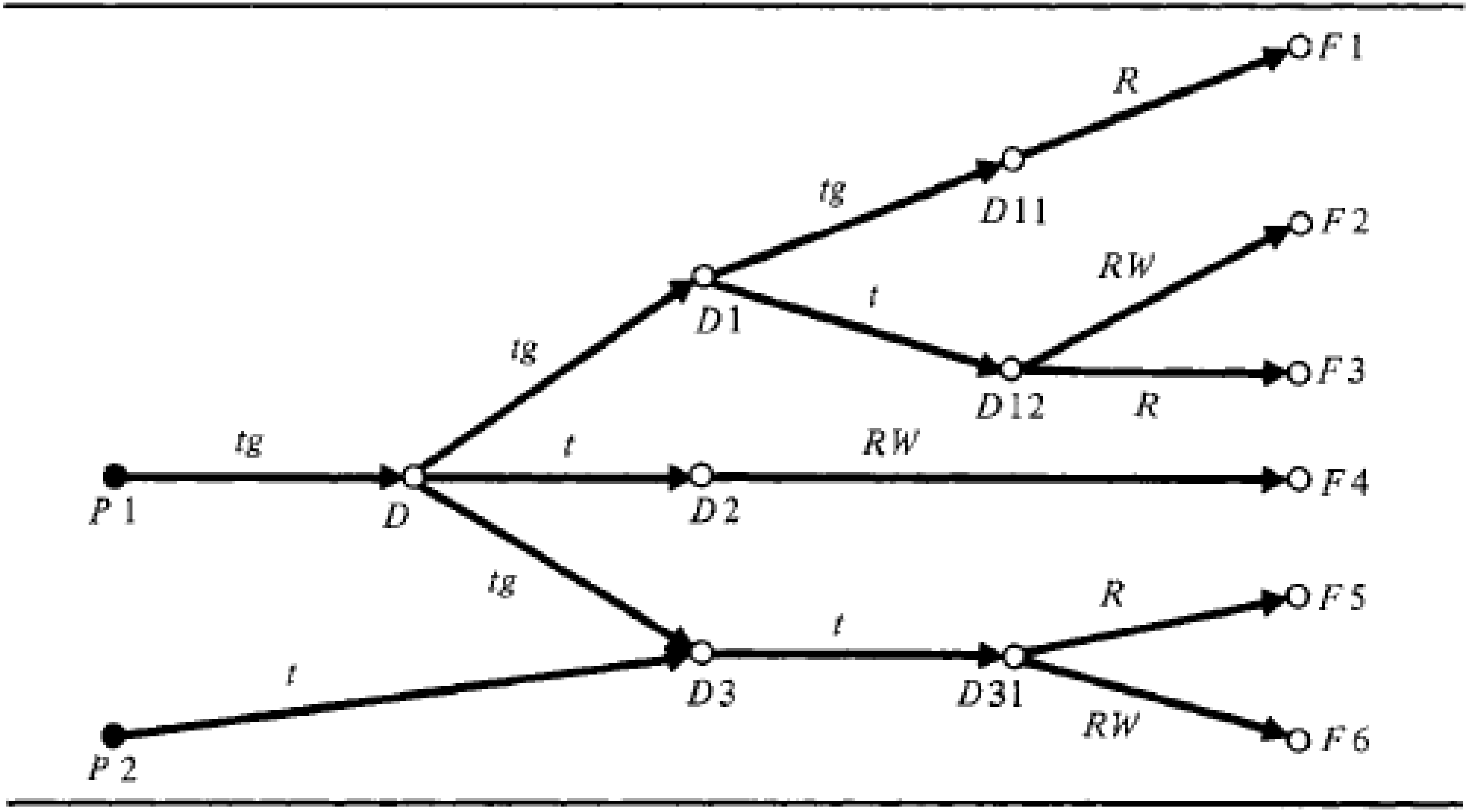
Shared subdirectory.



- P1 with an R W-capability for the root D of a file directory. To read F1, P1 would first get the R W-capability for directory D1 from D; it would then get the R W-capability for D II from D1, and finally get an R-capability for F1 from D11.
- Similarly,others
- Process P1 can share D3 with a P2 by giving it an R-capability for subdirectory D3;
- Thus P2 can only access files F5 and F6; it cannot modify any of the directories.

O – Objects; Black bullets: Subjects

Take-grant representation of directory structure.



Take Grant Model

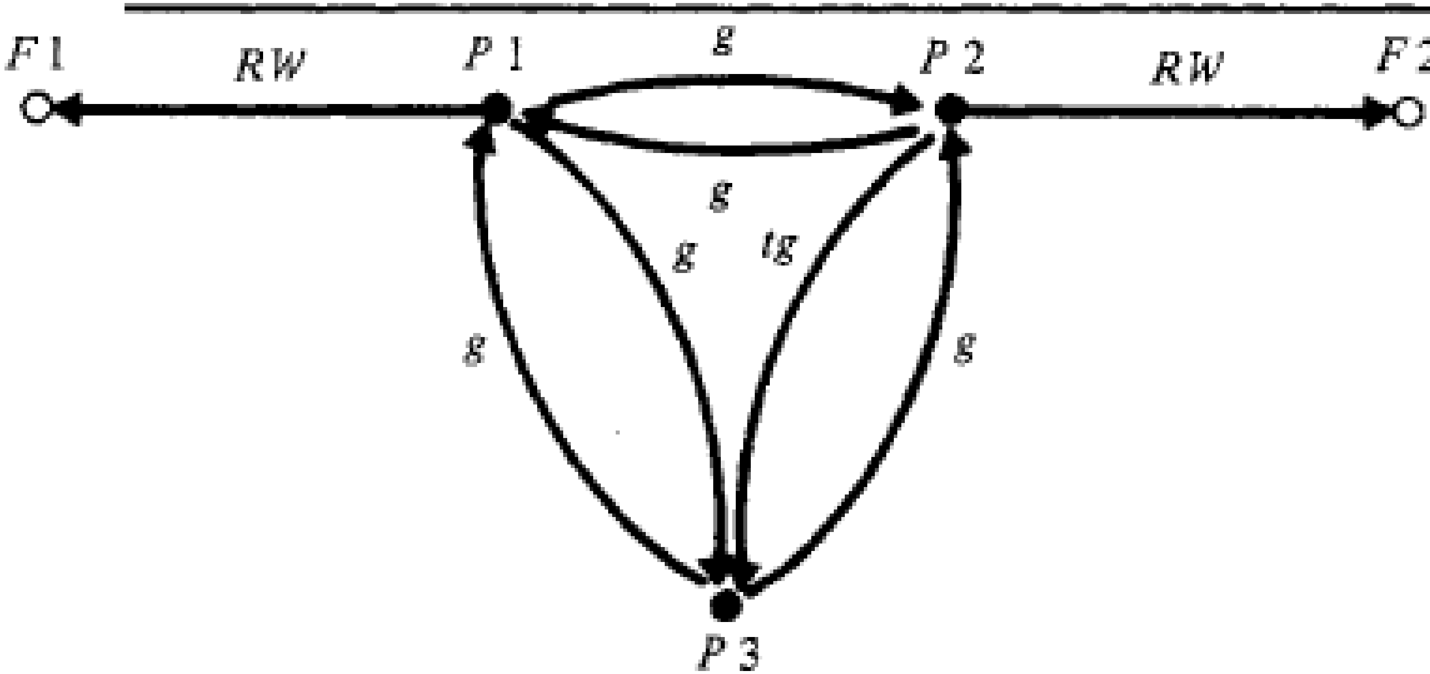
- The take-grant model describes the transfer **of authority (rights) in systems.**
- It does not describe the protection state with respect to rights that cannot be transferred.
- Thus, it abstracts from the complete state only information needed to answer questions related to safety.

Access matrix after command sequence.

		Objects							
		$M1$	$M2$	$M3$	$F1$	$F2$	$P1$	$P2$	$P3$
	$P1$	R W E			Own R W				
Subjects	$P2$		R W E	R W		Own R W			$Ctrl$
	$P3$			R W E		R			

Note: Process p is given the *Ctrl right to q, allowing it to take or revoke any of q'*
rights, including those conferred on q by other processes.

Take-grant graph for system shown in Figure 4.2.



- A process can grant rights for any of its owned files to any other process, so there is an edge labeled g connecting each pair of processes.
- Only process P2 is allowed to take rights from another process, namely its subordinate P3, so there is only one edge labeled t.

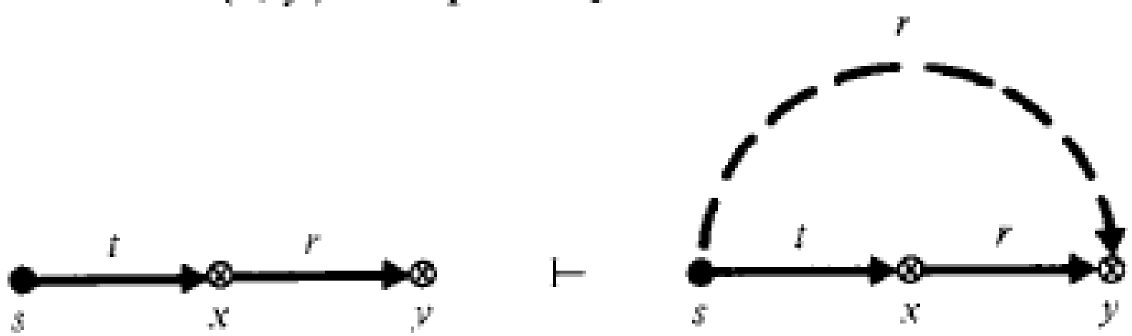
- Because rights for memory segments cannot be granted along the g-edges (memory is not owned and the copy flag is not set), these rights are not shown.
- Consequently, the graph does not show P2 can take P3's rights for memory segment M3 (as it did).

Graph Rewrite Rules (1)

1. **Take:** Let s be a subject such that $t \in (s, x)$, and $r \in (x, y)$ for some right r and nodes x and y. The command

   s **take** r for y from x

   adds r to (s, y). Graphically,



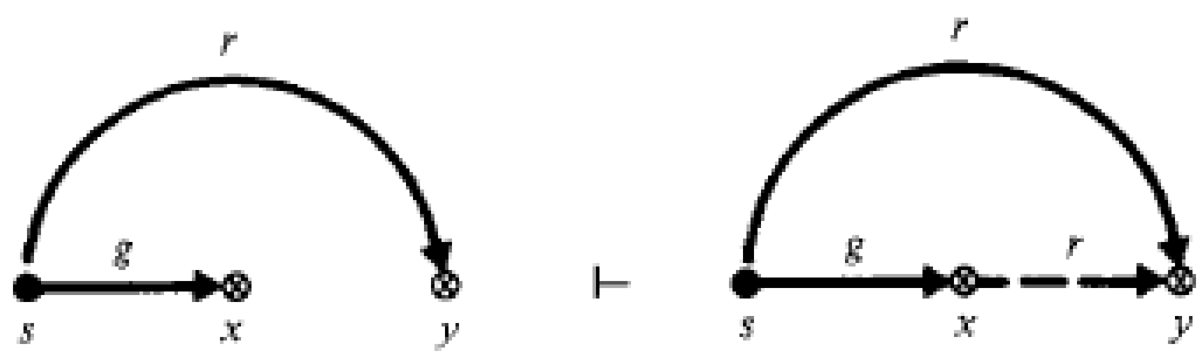
   where the symbol "⊗" denotes vertices that may be either subjects or objects.

Graph Rewrite Rules (2)

2. **Grant:** Let s be a subject such that $g \in (s, x)$ and $r \in (s, y)$ for some right r and nodes x and y. The command

s **grant** r for y to x

adds r to (x, y). Graphically,



Graph Rewrite Rules (3)

3. **Create:** Let s be a subject and ρ a set of rights. The command

$$s \ \textbf{create} \ \rho \ \text{for} \ \textbf{new} \begin{Bmatrix} \textbf{subject} \\ \textbf{object} \end{Bmatrix} x$$

adds a new node x and sets $(s, x) = \rho$. Graphically,

Graph Rewrite Rules (4)

4. **Remove:** Let s be a subject and x a node. The command

   s **remove** r for x

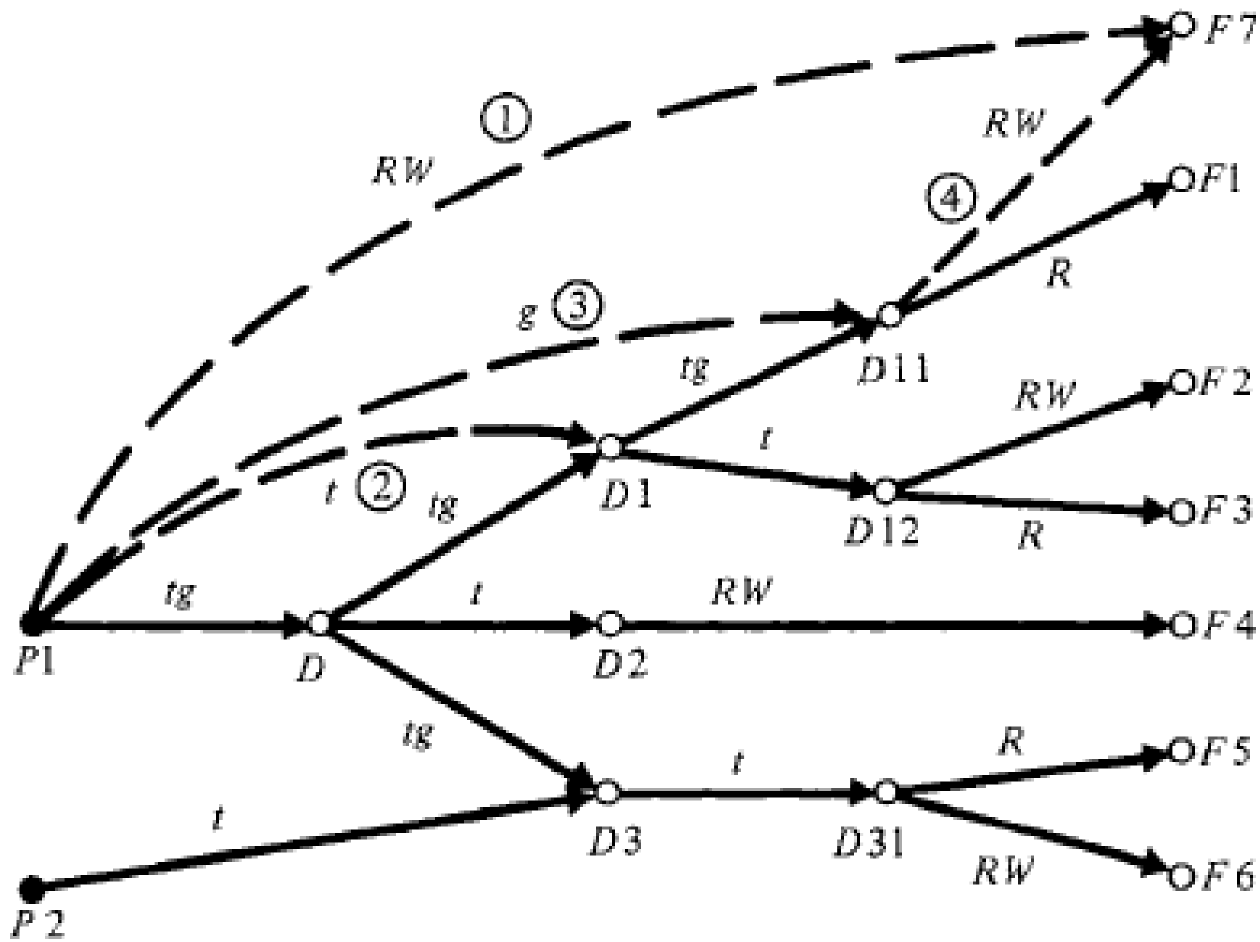
   deletes r from (s, x). Graphically,

$$s \xrightarrow{\rho} x \quad \vdash \quad s \xrightarrow{\rho - r} x$$

Example

- process P1 can create a new file F7 and add it to the directory D11
- P1 **create *RW for new object F7***
- P1 take t for D1 from D
- P1 **take g for D11 from D1**
- P1 grant *R W for F7 to D11*

Adding a new file $F7$ to directory $D11$.



Safety in Protection Graph

G : a protection graph.

- G |-- G' if command c transforms G into graph G',
- G |--c G' if there exists some command c such that G |--c G', and
- G |--* G' if there exists a (possibility null) sequence of commands that transforms G into G'.

Safety in Protection Graph

- Consider a node s with right r for node x; thus r in (s, x).
- Safety question: Determine whether another subject can acquire right r (not necessarily for x).
- Rather than consider whether an arbitrary subject can acquire the right r for an arbitrary node, consider whether a particular node p (subject or object) can acquire the particular right r for x.
- If this is decidable, so is the general question

Safety

- Given an initial graph Go with nodes s, x, & p such that r in (s, x) and r not-in (p, x), Go is safe for the right r for x
- if and only if r not-in (p, x) in every graph G derivable from Go (i.e., Go |--* G).
- Proof in two parts:
  - s can "share" its right with other nodes (but not necessarily p)
  - the right must be "stolen" from s.

Proof

- Go with nodes p & x s.t. r not-in(p, x) in Go
- predicate *can.share(r, x, p, Go) is true iff there exists a node s in Go such* that r in (s, x) &
- Go is unsafe for the right r for x; that is, p can acquire the right r for x.
- Rights can only be transferred along edges labeled either t or g.
- Nodes x and y are tg-connected if there is a path between them s.t. each edge on the path is labeled with either t or g (direction is not important);
- Directly tg-connected if the path is just (x, y) or (y, x).

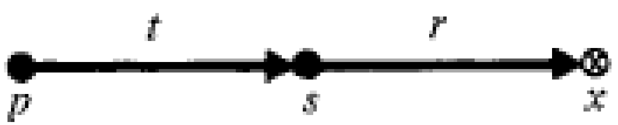
Proof of *canshare*

Theorem1:

- *can.share(r, x, p Go) is true if p is a subject and*
- There exists a subject s in Go such that r in (s, x) in Go, and
- s and p are directly tg-connected.

Proof of canshare - Theorem

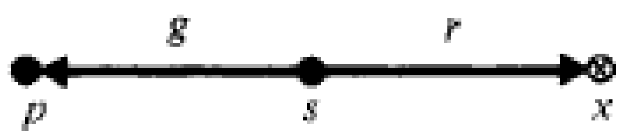
Case 1.



The first case is simple, as p can simply take the right r from s with the command:

p **take** r for x from s

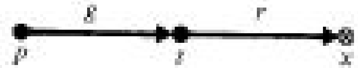
Case 2.



This case is also simple, as s can grant (share) its right to p with the command:

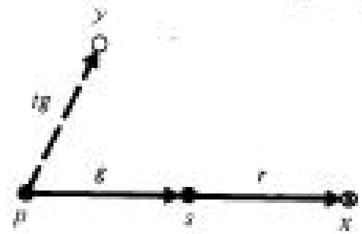
s **grant** r for x to p

Case 3.

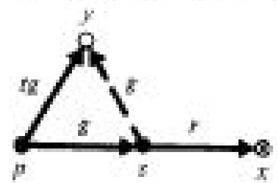


This case is less obvious, as p cannot acquire the right with a single command. Nevertheless, with the cooperation of s, p can acquire the right with four commands:

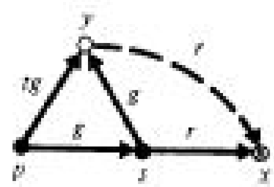
p **create** tg for **new object** y



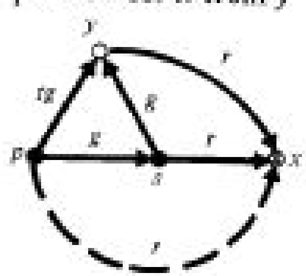
p **grant** g for y to s



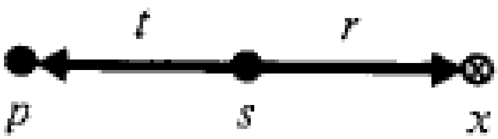
s **grant** r for x to y



p **take** r for x from y



Case 4.



This case also requires four commands; we leave it as an exercise for the reader. ■

This result is easily extended to handle the case where subjects s and p are *tg-connected* by a path of length >=1 consisting of subjects only.

Letting p = P0, P1 . . . , pn= s denote the path between p and s, each p can acquire the right from *Pi+l as described in Thm 1.*

It turns out that tg-connectivity is also a necessary condition for *can.share in graphs containing only* subjects

Theorem 2 :

If *Go is a subject-only graph,*

then can.share(r, x, p, Go) is true if f:

a. There exists a subject s in Go such that r in (s, x), and

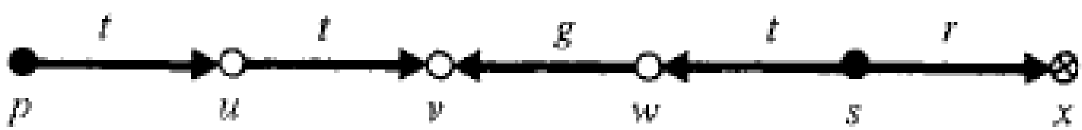
b. s is tg-connected to p.

- Graph Contains both Subjects and Objects

An **island** is any maximal subject-only *tg*-connected subgraph. Clearly, once a right reaches an island, it can be shared with any of the subjects on the island. We must also describe how rights are transferred between two islands.

A ***tg*-path** is a path $s_1, o_2, \ldots, o_{n-1}, s_n$ of $n \geq 3$ *tg*-connected nodes, where s_1 and s_n are subjects, and $o_2, \ldots, o_{n-1}$ are objects. A ***tg*-semipath** is a path $s_1, o_2, \ldots, o_n$ of $n \geq 2$ *tg*-connected nodes, where s_1 is a subject, and $o_2, \ldots, o_n$ are objects. Each *tg*-path or semipath may be described by a word over the alphabet $\{\overrightarrow{t}, \overrightarrow{g}, \overleftarrow{t}, \overleftarrow{g}\}$.

Example:
The *tg*-path connecting p and s in the following graph

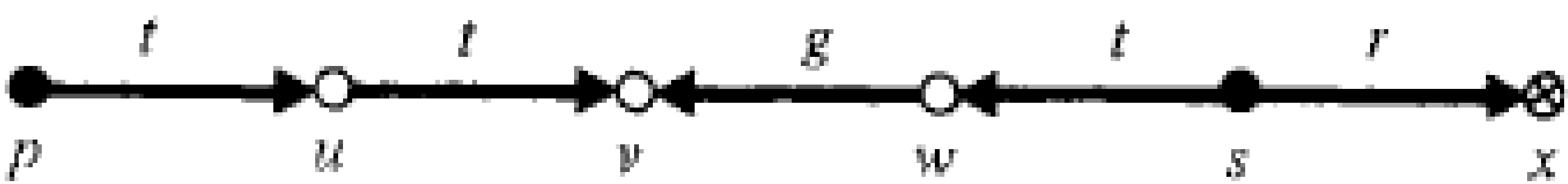


is described by the word $\overrightarrow{t}\,\overrightarrow{t}\,\overleftarrow{g}\,\overleftarrow{t}$. ■

A **bridge** is a *tg*-path with an associated word in the regular expression:

$$(\overrightarrow{t})^* \cup (\overleftarrow{t})^* \cup (\overrightarrow{t})^* \overrightarrow{g} (\overleftarrow{t})^* \cup (\overrightarrow{t})^* \overleftarrow{g} (\overleftarrow{t})^* \ .$$

Bridges are used to transfer rights between two islands.



$$\overrightarrow{t} \quad \overrightarrow{t} \quad \overleftarrow{g} \quad \overleftarrow{t}$$

How s can share its right r for x with p ?

An **initial span** is a tg-semipath with associated words in

$(\vec{t})^* \ \vec{g}$

and a **terminal span** is a tg-semipath with associated word in

$(\vec{t})^*$.

- Arrows emanate from the subject sl in the semipaths.

•Note that a bridge is a composition of initial and terminal spans.
The idea is that a subject on one island, is responsible for transferring a right over the initial span of a bridge, and

a subject on the other island is responsible for transferring the right over the terminal span;

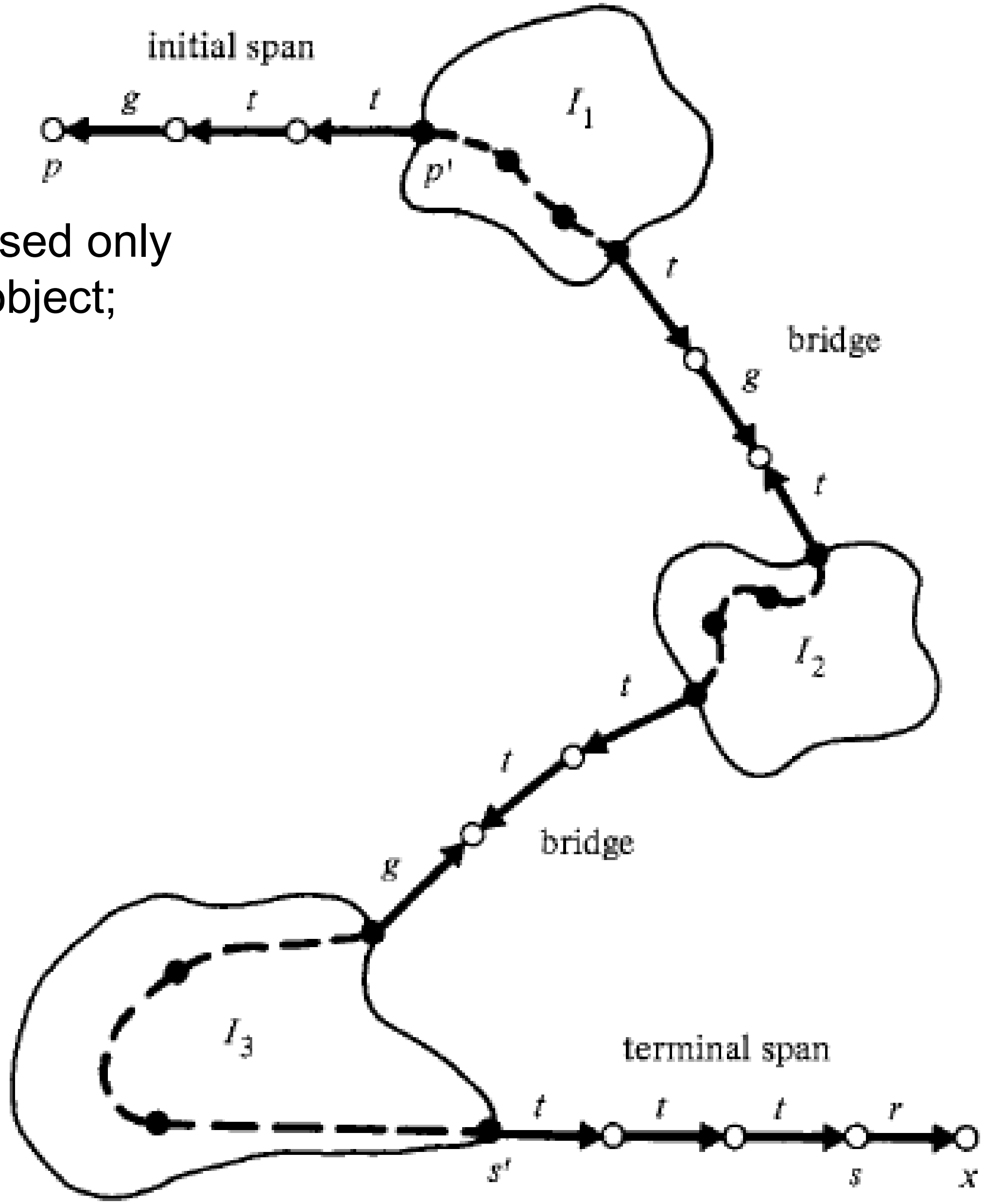
the middle of the bridge represents a node across which neither subject alone can transfer rights.

Main Theorem

- *Thm **3: P**redicate can.share(r, x, p, Go) is true if and only if:*
  - There exists a node s such that r in (s, x) in Go; and There exist subjects p' and s' such that
    - p' = p (if p is a subject) or p' is tg-connected to p by an initial span (if p is an object), and
    - s' = s (if s is a subject) or s' is tg-connected to s by a terminal span (if s is an object); and
    - There exist islands I1 . . . , Iu (u > =1) such that p' in I1, *s' in I,, and* there is a bridge from Ij to Ij+l

      (1<= j < u)

Path over which r for x may be transferred from s to p.



initial span is used only when p is an object;

A terminal span is similarly used only when s is an object

Algorithm

- There is an algorithm for testing *can.share that operates in linear time in the* size of the initial graph,

Stealing

- A node p steals a right r for x from an owner , s if it acquires r for x without the explicit cooperation of s.
- I.e., *can.steal(r, x, p, Go) is true iff p does not have r* for x in Go and there exist graphs G1, . . . , Gn, such that:

1. *Go l—c1 G1 l—c2 . . . L--cn Gn;*
2. r in (p, x) in Gn; and
3. For any subject s such that r in (s, x) in Go, no command *ci is of the form*

   s **grant r for x to y**

   for any node y in Gi-1.

Theorem

- ***Theorem 4:*** *can.steal(r, x, p, Go) is true iff*

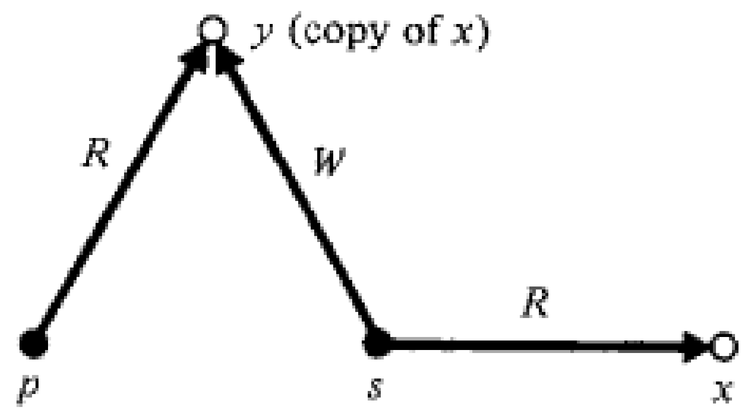
1. There is a subject p' such that p' = p (if p is a subject) or p' initially spans to p (if p is an object), and
2. There is a node s such that r in (s, x) in Go and *can.share(t, s, p', Go) is* true; i.e., p' can acquire the right to take from s.

- I.e., if a subject cannot acquire the right to take from s, then it cannot steal a right from s by some other means.
- Stealing Subjects are called conspirators.

If a subject cannot steal a right for an object x, this does not Necessarily mean the information in x is protected

Eg, Another subject s may copy the information in x into another object y that p can read

De facto acquisition.



T-G System

- it does describe many aspects of existing systems, especially capability systems.
- Nevertheless, the results are significant because they show that in properly constrained systems, safety decisions are not only possible but relatively simple.
- Safety is undecidable in HRU because the commands of a system were unconstrained;
  - a command could, if desired, grant some right r for x to every subject in the system.
- The take-grant model, on the other hand, constrains commands to pass rights only along tg-paths.

Access Matrix Vs Take Grant

Properties that can be Analysed by ACM but not by T-G

- The strength of the Access Control Matrix Model – flexibility.
- Unlike the TG, no condition on the types of rights or their flow between subjects & objects, & hence can model a wider variety of protection systems.
- Consequences of destruction of entire objects can be studied only in ACM, since TG does not permit the operations
- TG: is not expressive enough to study change in protection state over parameters such as time of the day and role of the subject

Properties that can be Analysed by TG but not by ACM

- Is the ability to analyse the safety of a protection system in time linear with the number of subjects and objects in the system.

- In case system safety is compromised wrt right r, TG allows us to analyse all possible subjects that needto be involved in order to leak it.
- even possible to analyse whether a theft of that is possible by a subject without needing coop.of other subject
- Enable us to quantify the amount of trust that the system would be placing on different subjects.