Faster Operation in GARP

BROADEN YOUR LIFE

Jessy V Rouyer
Oscar Rodriguez
Kamakshi Sridhar

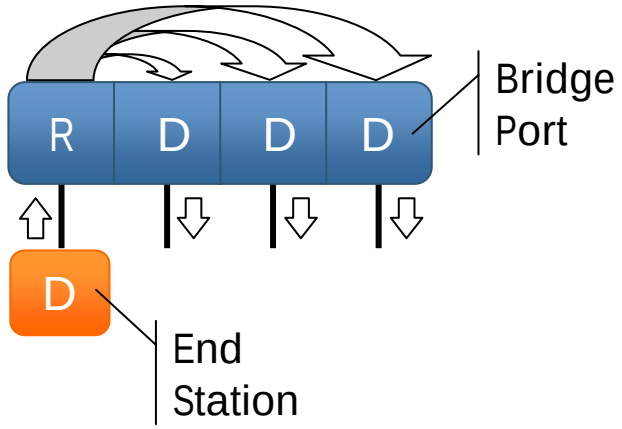
ALCATEL

Agenda

- GARP basics
- GARP weaknesses
  - Slow convergence
  - Slow processing
- Modifications for faster operation in GARP
  - Faster convergence
  - Faster processing
- Conclusion

GARP basics (1/2)

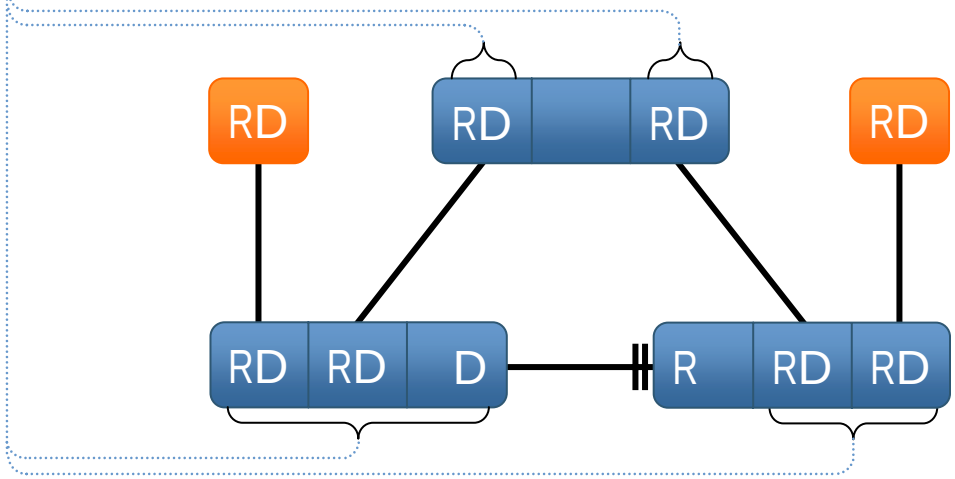
- The Generic Attribute Registration Protocol (GARP):
  - **Automatically registers, maintains, de-registers, disseminates attributes**
  - Supports GARP-based applications for specific attributes (e.g. GVRP, GMRP)
- The GARP Information Propagation Component (**GIP Component**) propagates attributes between ports participating in a same application:
  - An end station Declares (D) an attribute
  - The bridge port connected to this end station Registers (R) this attribute
  - This attribute is propagated by the GIP Component and Declared (D) on other participating bridge ports
  - Those bridge ports eventually propagate this attribute over the active topology



- GARP state machines **periodically transmit attribute information** over the active topology defined by the running R/M/STP

GARP basics (2/2)

> The **GIP Context** is the set of ports belonging to the **active topology**



> The addition or removal of a port from the GIP Context can be either:
  - Controlled (e.g. via management)
  - Uncontrolled (e.g. after a topology change)

> GARP tries to minimize control traffic to preserve bandwidth
  - **Transmits periodically relying on 3 timers**

3 timers	Min	Max
Join	0 ms	200 ms
Leave	600 ms	
LeaveAll	10 s	15 s

GARP weaknesses

3 timers	Min	Max
Join	0 ms	200 ms
Leave	600 ms	
LeaveAll	10 s	15 s

- GARP weaknesses:
  - Convergence:

	When?	Why?
Slow	New Declaration	Dominated by **Join**
Very Slow	Withdrawn Declaration	Dominated by **Leave**
Extremely Slow	Topology Change	Dominated by **LeaveAll**
Incorrect	Removed Port	Deficiency in GIP

  - Processing: slow on bridges with many ports

Modifications for Faster Operation in GARP

- We propose **modifications to GARP** that overcome its weaknesses by:
  - Speeding up GARP convergence
  - Reducing GARP processing
  - How? By making GARP more proactive (à la RSTP)

    With **NO modification to GARP timer values**

    With **NO modification to GARP state machines**

    With **NO significant increase in GARP control traffic**

    Yet **maintaining backward compatibility with GARP**

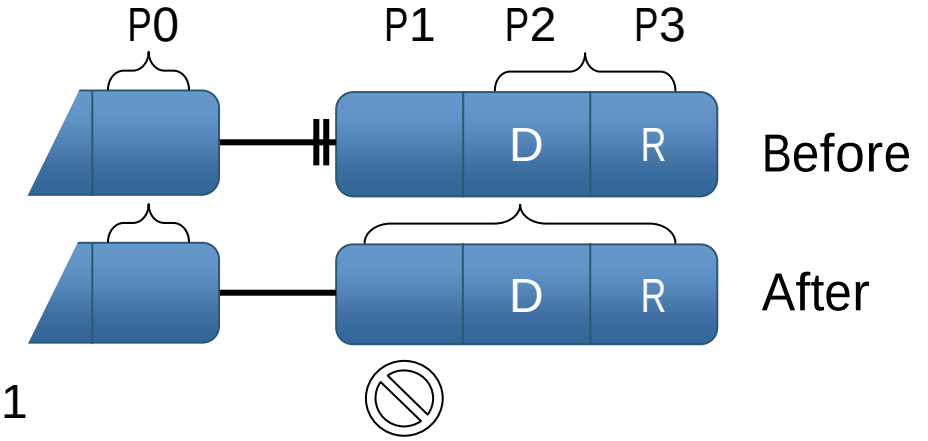
Modifications for Faster Operation in GARP

Speeding up GARP convergence

Propagating registered attributes to a port when adding the port to the GIP Context

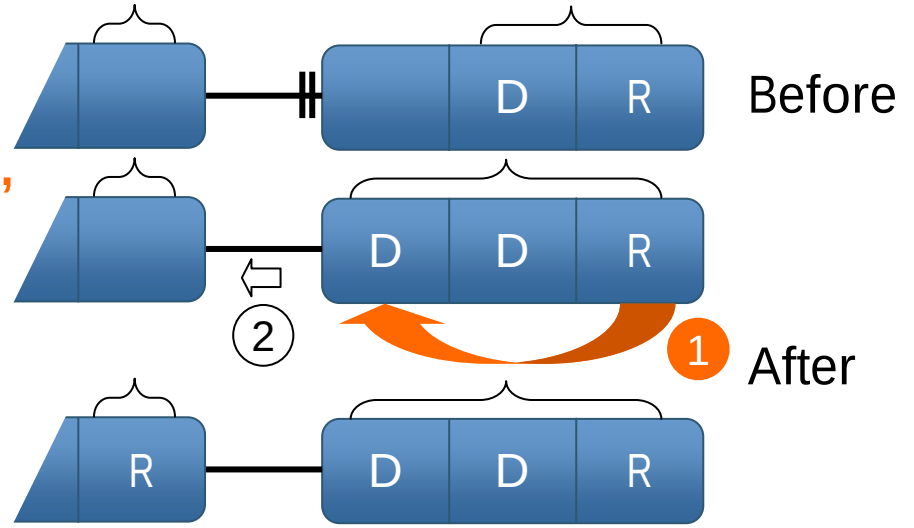
> With **GARP**:

- P1, when added to the GIP Context, is not aware of attribute(s) registered on other port(s)
- P1 has to wait for **LeaveAll timer** to expire on P3 (up to 15+s) before P3 propagates GID_Join.request(s) to P1



> **Proposed modification:**

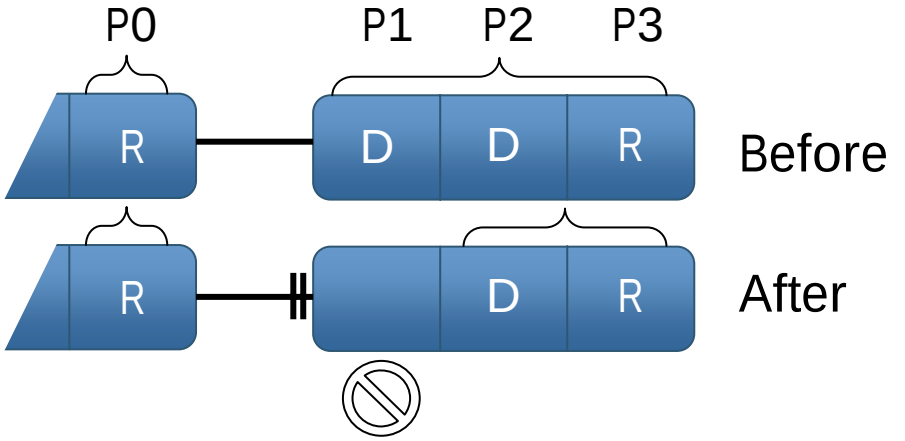
1. **P1, when added to the GIP Context, receives GID_Join.request(s) immediately propagated by other port(s) with registered attribute(s)**
2. As a consequence, P0 can register the attribute(s) declared by P1 faster



Transmitting a LeaveEmpty message when removing a port from the GIP Context

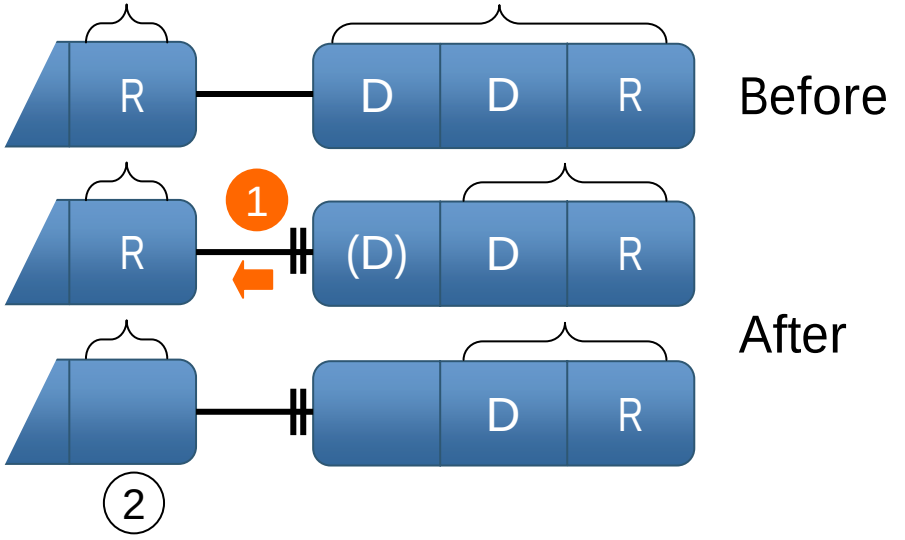
> With **GARP**:

- P1, when removed from the GIP Context, does not inform P0 that it should drop registered attribute(s)
- P0 has to wait for its **LeaveAll timer** to expire (up to 15+s) before it can drop its registered attribute(s)



> **Proposed modification**:

1. **P1, when removed from the GIP Context, transmits a LeaveEmpty message for each attribute declaration that it made**
2. As a consequence, P0 can drop its registered attribute(s) faster



Deficiency in GIP when removing a port from the GIP Context (1/2)

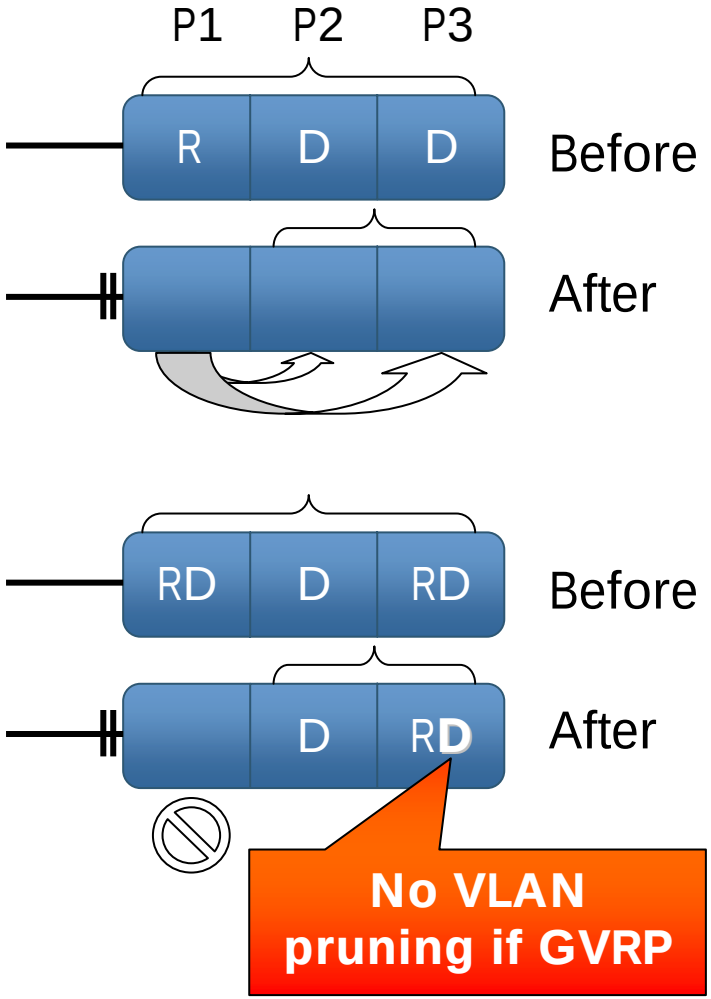
> With **GARP**:

- "If a port is removed from the set, and that port has registered an attribute and **no other port** has, then GID_Leave.requests **are** propagated to the GID instances for each of the other Ports in the set." (IEEE Std 802.1D-2004 p82)

This is correct

- However: if a port is removed from the set, and that port has registered an attribute and **one other port** has, then GID_Leave.requests **are not** propagated to the GID instances for each of the other Ports in the set.

This is not correct



> **Incorrect convergence**

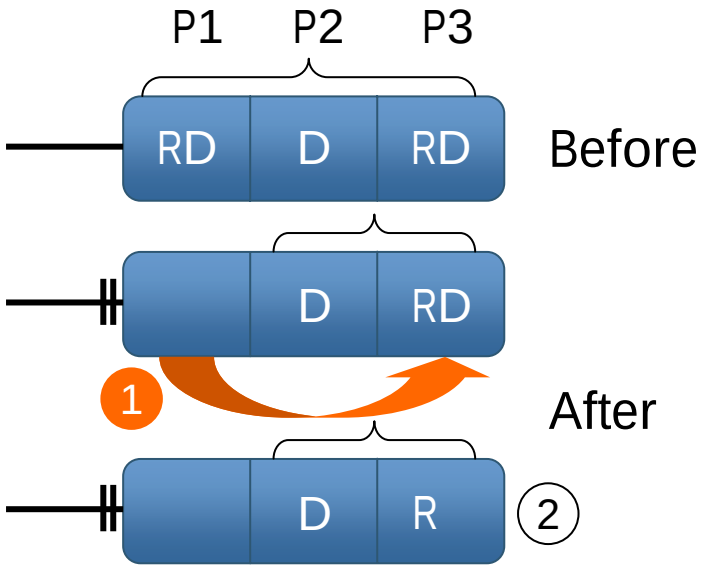
Deficiency in GIP when removing a port from the GIP Context (2/2)

> Propagation of GID_Leave.requests to other port(s):

	Standard	Proposal
No other registration	**Yes**	**Yes**
1 other registration	**No**	**Yes**
>1 other registration	**No**	**No**

> **Proposed modification:**

1. • Add:
   if a port is removed from the set, and
   that port has registered an attribute and
   one other port has, then GID_Leave.requests
   are propagated to the GID instances
   for **that other Port**.
2. • Declaration removed and propagated



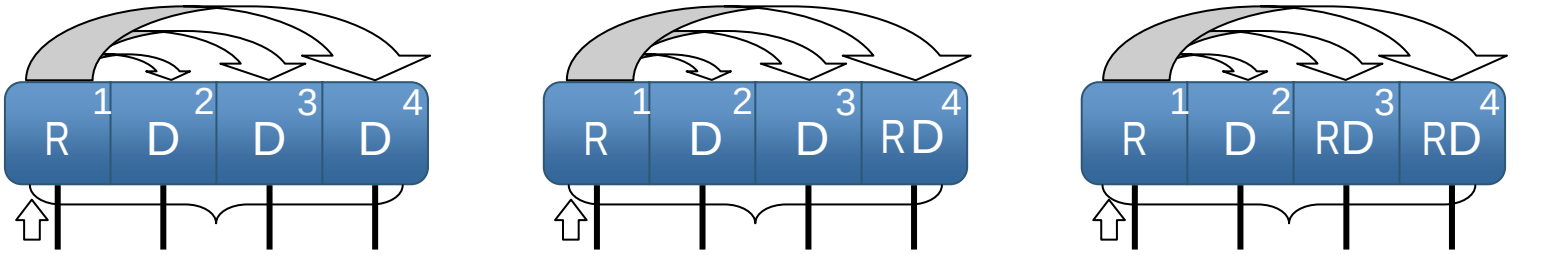
> **Correct convergence**

Modifications for Faster Operation in GARP

Reducing GARP processing

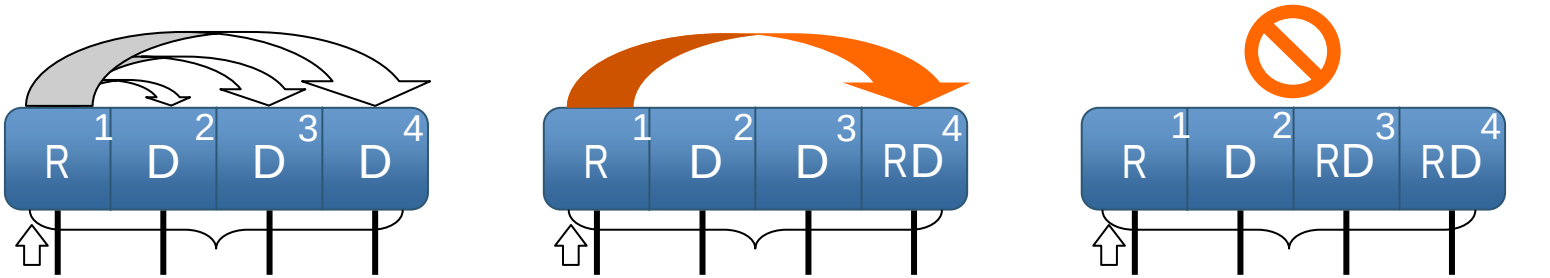
Selectively propagating GID_Join.requests when registering an attribute

- When registering an attribute, GARP propagates GID_Join.requests to all other ports regardless of existing registrations on those other ports



No other registration | 1 other registration | >1 other registration

- **Proposed modification: selectively propagate GID_Join.requests**



No other registration | 1 other registration | >1 other registration

- **Faster processing** by avoiding unnecessary propagations

Conclusion

- We propose modifications to speed up GARP:
  - Offering **faster convergence, faster processing**
  - With **NO modification to GARP timer values**
  - With **NO modification to GARP state machines**
  - With **NO significant increase in GARP control traffic**
  - That **maintain backward compatibility with GARP**
  - That **are tested through simulations**

www.alcatel.com

ALCATEL