



max planck institut
informatik

Formal verification of Pastry Using TLA+

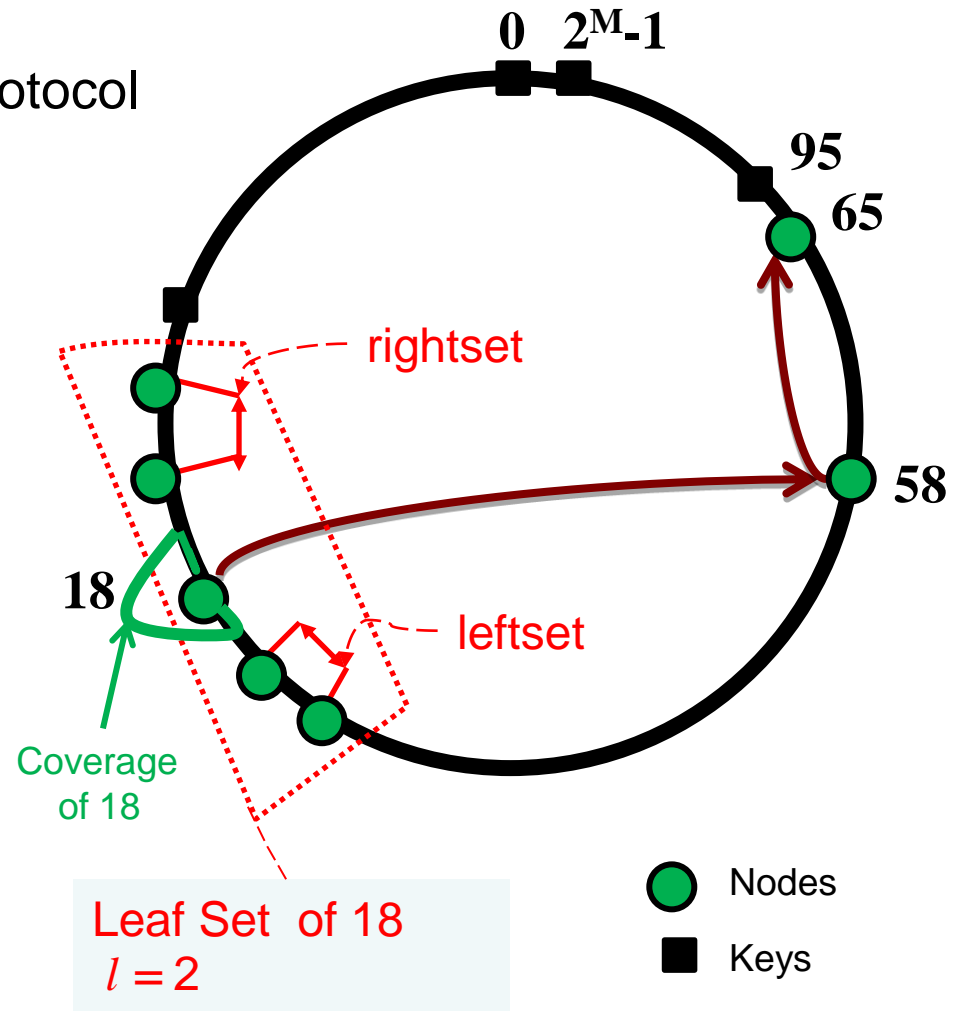
Tianxiang Lu
Stephan Merz
Christoph Weidenbach

TLA+ Workshop at FM2012, Paris

August 27, 2012

Introduction

- Pastry
 - Overlay P2P network protocol
 - Distributed Hash Table
 - Self organized nodes
 - Resilient to churn:
 - concurrent join
 - silent departure
- Virtual ring
 - (see the picture)



Introduction

- Verification Challenges
 - Complex data structure
 - Distributed protocol: absence of global state
 - Dynamic network: spontaneous departure, join of nodes
- Today I will talk about
 - How we formally modeled Pastry in TLA⁺
 - How we prove properties of Pastry using TLAPS

Formal Model in TLA+

$$\begin{aligned} \text{vars} &\triangleq \langle \text{receivedMsgs}, \text{status}, \text{lset}, \text{probing}, \text{failed}, \text{rtable} \rangle \\ \text{Init} &\triangleq \wedge \text{receivedMsgs} = \{\} \\ &\wedge \text{status} = [i \in I \mapsto \text{IF } i \in A \text{ THEN "ready" ELSE "dead"}] \\ &\wedge \text{probing} = [i \in I \mapsto \{\}] \\ &\wedge \text{failed} = [i \in I \mapsto \{\}] \\ &\wedge \text{lset} = \dots \\ &\wedge \text{rtable} = \dots \\ \text{Next} &\triangleq \exists i, j \in I : \vee \text{RouteLookup}(i, j) \vee \text{RouteJoin}(i, j) \vee \text{Deliver}(i, j) \\ &\vee \text{Join}(i, j) \vee \text{JReply}(j, i) \vee \text{Probe}(i) \vee \text{PReply}(i, j) \\ &\vee \text{NodeLeft}(i) \vee \text{SuspectFaulty}(i, j) \vee \text{ProbeTimeOut}(i, j) \\ &\vee \text{LSRepair}(i) \vee \text{Recover}(i) \vee \text{ResignNode}(i) \\ &\vee \text{RequestLease}(i) \vee \text{ReceiveLReq}(i) \vee \text{ReceiveBLS}(i) \\ &\vee \text{LeaseExpired}(i, j) \vee \text{DeclareDead}(i, j) \\ \text{Spec} &\triangleq \text{Init} \wedge \square[\text{Next}]_{\text{vars}} \end{aligned}$$

Verification Target

- Validate model by refuting impossibility claims
 - *NeverJoin*: A new node can never be joined the network
 - *NeverDeliver*: A lookup message can never be delivered
- Safety Property: Correct Delivery
 - For each key k , there is at most one node i that may deliver, and no other node is closer to k than i .

$CorrectDelivery \triangleq \forall i, k \in I :$

$ENABLED \ Deliver(i, k)$

$\Rightarrow \wedge \forall n \in I : status[n] = \text{“ready”} \Rightarrow AbsDist(i, k) \leq AbsDist(n, k)$

$\wedge \forall j \in I \setminus \{i\} : \neg ENABLED \ Deliver(j, k)$

Model Checking Pastry Properties

- Model Checking using TLC
- Statistics
 - 8 state variables
 - 11 concurrent actions
 - Total state space roughly: $2^{152} \times 3^{64}$ ($\approx 10^{76}$) for 4 nodes
 - Server with 2 CPUs (32 Bit Linux machine with Xeon(R) X5460)
 - 3.16GHz, 4 GB of memory per CPU

Property	Time	Depth	# states	Counter Example
NeverDeliver	1"	5	101	yes
NeverJoin	1"	9	19	yes
.....				
CorrectDelivery	> 1 month	21	1952882411	no

Proving Correct Delivery

- To prove: $Spec \Rightarrow \Box CorrectDelivery$

1. Invent a property *Inv*, in order to apply the rule

$$\frac{Spec \Rightarrow \Box Inv \quad Inv \Rightarrow CorrectDelivery}{Spec \Rightarrow \Box CorrectDelivery}$$

2. Prove $Spec \Rightarrow \Box Inv$ by:

$$\frac{Init \Rightarrow Inv \quad Inv \wedge A(i, j) \Rightarrow Inv' \text{ for every sub-action } A(i, j) \text{ of } Next}{Spec \Rightarrow \Box Inv}$$

- Recall that $Spec = Init \wedge \Box [Next]_{vars}$

Proof in TLA+ toolbox

- Proof of the model in TLAPS with strong assumptions
 - no nodes leave the network
 - only one node can join the network at a time in any neighboring region
- Statistics
 - 23 invariants proved by induction on 11 actions
 - About 100 lemmas on arithmetic and ring calculation
 - About 100 lemmas on data structures
 - About 1200 proof steps for proving type correctness
 - About 12500 proof steps for inductive proof of invariants
 - CPU Intel Core i3-2330M 2.20GHz, 8 GB RAM, 64-bit, Win7
 - JVM `-Xms5120M -Xmx5120M -XX:PermSize=2048M`
 - About 10 minutes and 5GB for generating proof obligations

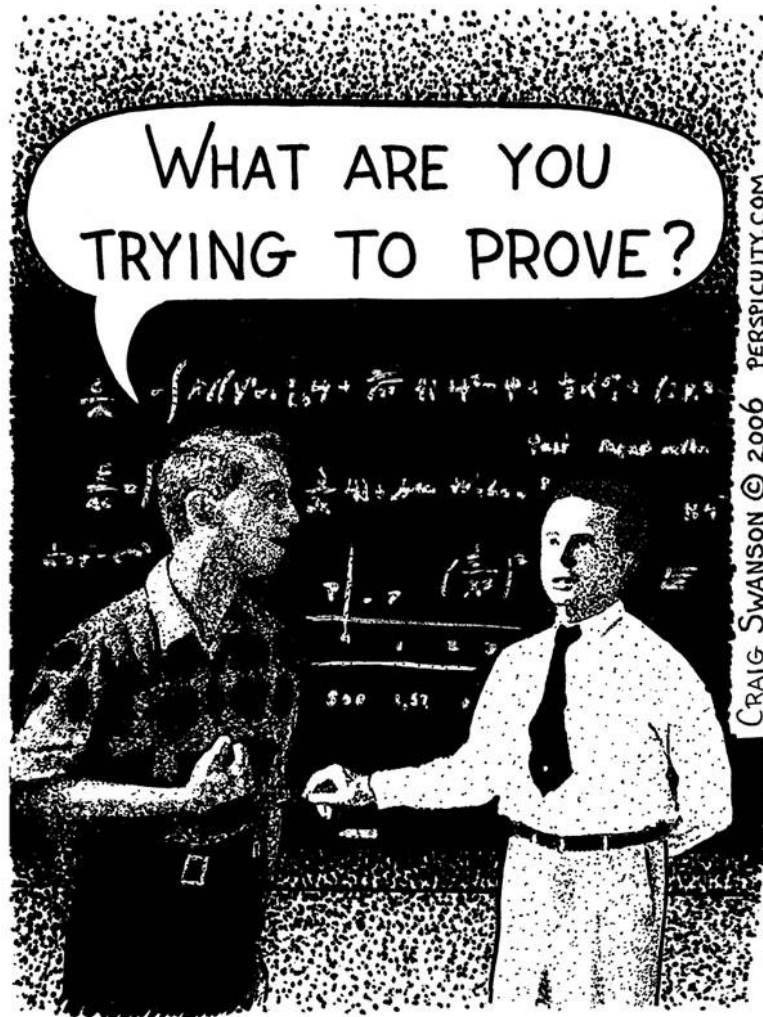
Done & Doing

- Done
 - Real-world case study of complex network protocol: Pastry
 - Found bugs in Protocol and improved it.
 - Modeled routing and join protocols in TLA+ and model checked them in TLC
 - Finished the proof of the model in TLAPS with strong assumptions
- Doing
 - Relaxed the assumptions: more nodes join in neighboring region
 - Finding the proper invariants and proving them

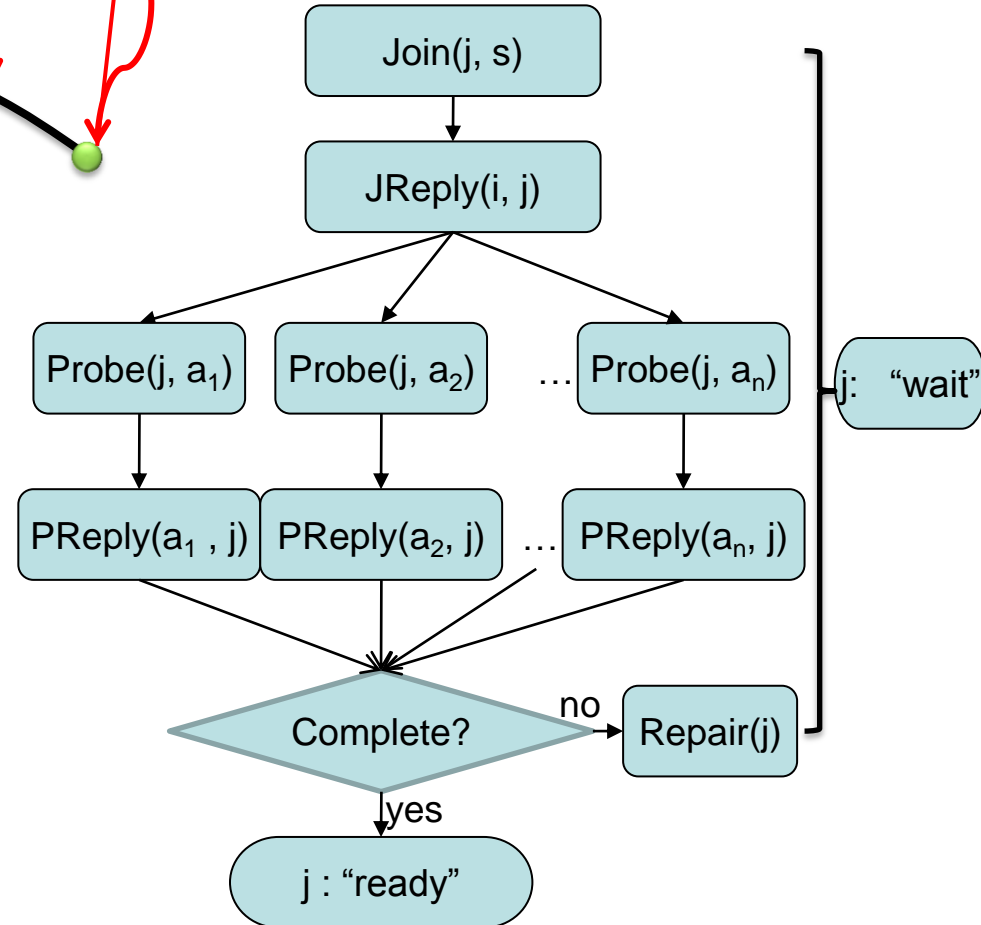
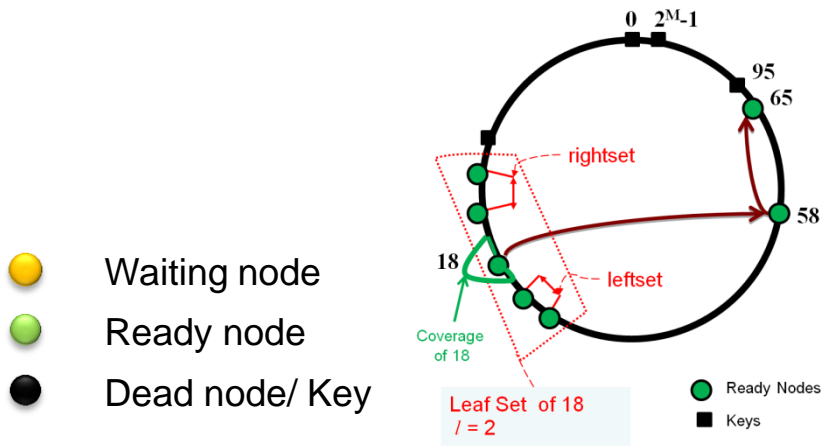
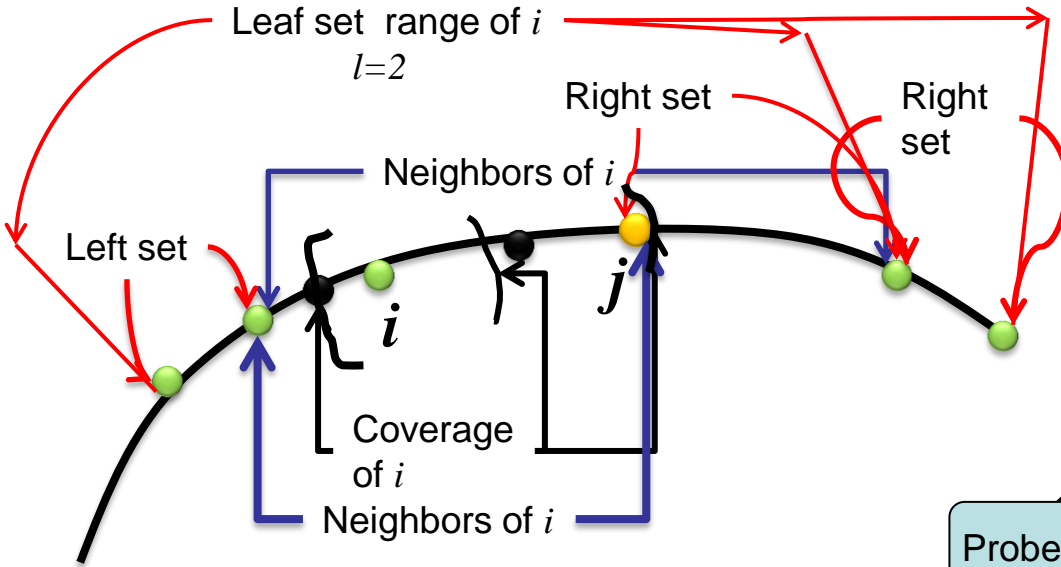
Remarks on the Tools

- Trace explorer
 - Very useful !
 - Display the action name ?
- TLC with multi-threads
 - Significant speed up
 - Huge memory footprint and no CPU usage after weeks
 - Java runtime problem ?
 - What about distributed version of TLC ?
- TLAPS
 - Proof editing is very convenient! (zoom, non-linear, jump ...)
 - Generation of proof obligation caused memory problem ?

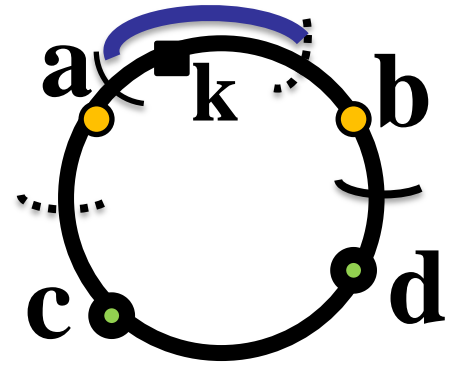
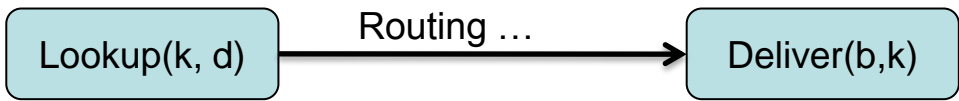
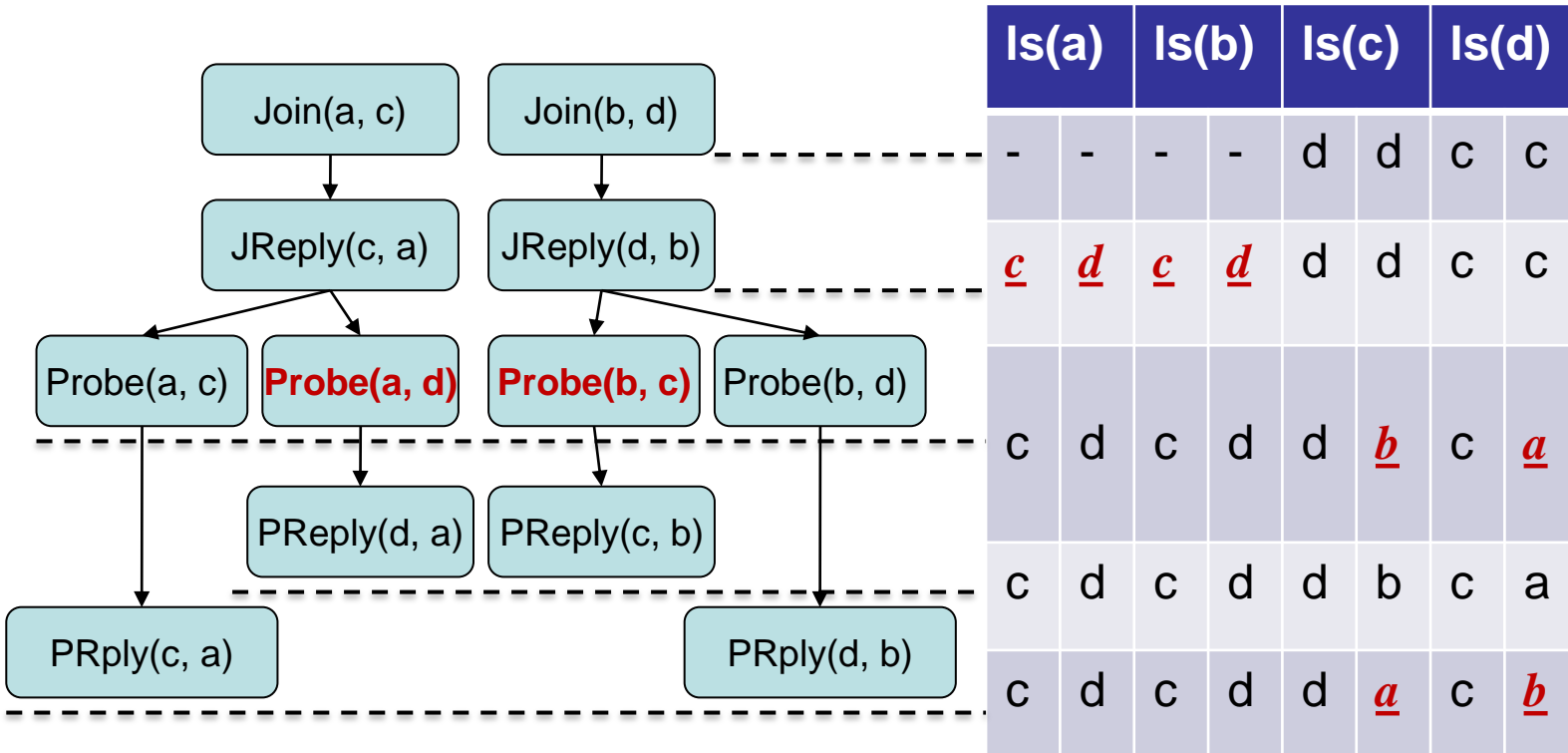
Thank you !



Join



Bug of Pastry



Lease Granting Protocol

[Haeberlen et al. 2005, FreePastry]

