COS 598D:

PATTERNS IN NETWORK ARCHITECTURE

Pamela Zave

AT&T Labs—Research and Princeton University

Spring 2017, Tuesdays 1:30 - 4:20, CS 402

http://www.cs.princeton.edu/courses/archive/spr16/cos598D

PATTERNS IN NETWORK ARCHITECTURE

TODAY'S OUTLINE

What is this course about?

How the course will be conducted

Introduction to a compositional model of networking

How to read a paper

Introduction to lightweight modeling in Alloy (net1.als)

WHAT IS A NETWORK ARCHITECTURE?

WHAT IS A NETWORK ARCHITECTURE?

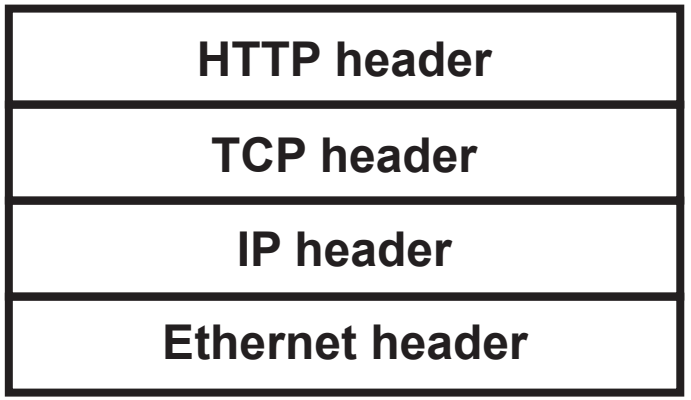
A DESCRIPTION OF A SET OF SIMILAR NETWORKS, INCLUDING:

- component types and their attributes
- relations among components
- division of labor among components
- other constraints

THE "CLASSIC" INTERNET ARCHITECTURE

APPLICATION LAYER	applications and mnemonic names
TRANSPORT LAYER	reliable byte streams, messages
NETWORK LAYER	best-effort global packet delivery
LINK LAYER	best-effort local packet delivery
PHYSICAL LAYER	many physical media (wires, optical fibers, radio channels)

so we would expect a typical packet to look like this



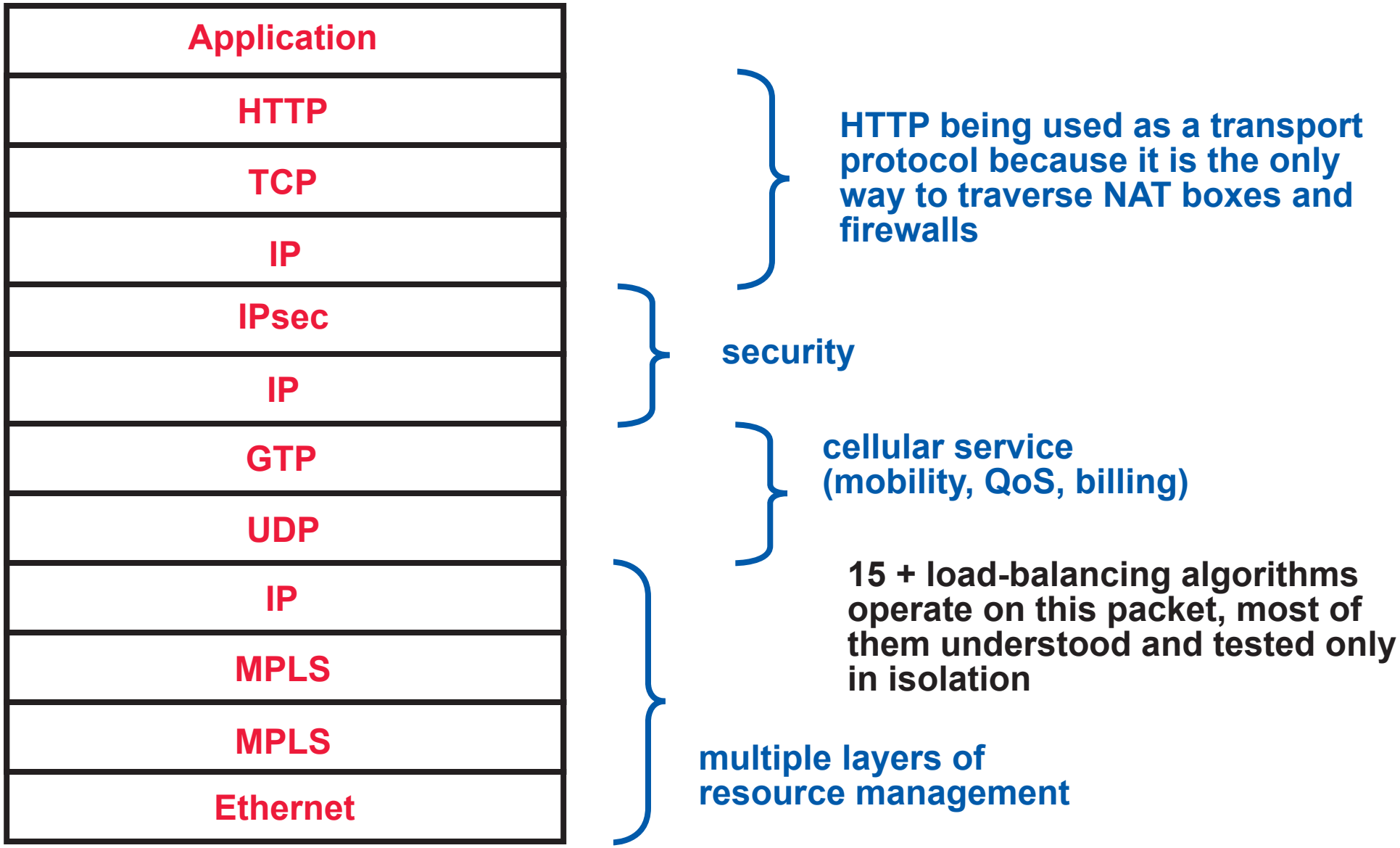
WHAT HAS HAPPENED SINCE 1995

- most of the world's telecommunication infrastructure has moved to the Internet
- much of the world's entertainment distribution has moved to the Internet
- an explosion of new networked devices and edge communication technologies
- mobility is king
- an explosion of security threats
- cloud computing
- exhaustion of the IP address space
- the need for elastic resource allocation instead of over-provisioning

almost every change makes demands that the classic Internet architecture cannot satisfy

A REAL PROTOCOL STACK

headers in a typical AT&T packet (12 instead of 4)



“There is a tendency in our field to believe that everything we currently use is a paragon of engineering, rather than a snapshot of our understanding at the time. We build great myths of spin about how what we have done is the only way to do it . . . to the point that our universities now teach the flaws to students . . . who don’t know better.”

— John Day

WHAT IS A DESIGN PATTERN?

WHAT IS A DESIGN PATTERN?

A RE-USABLE SOLUTION FOR A SET OF WELL-KNOWN PROBLEMS

patterns are very useful even if informal

this course is named for John Day's 2008 book

Patterns in Network Architecture

but it goes much, much further

and will be (hopefully)

much, much clearer

WHAT IS A MODEL?

WHAT IS A MODEL?

Dictionary.com:

1. **a standard or example for imitation or comparison**
2. **a representation, generally in miniature, to show the construction or appearance of something**
3. **an image in clay, wax, or the like, to be reproduced in more durable material**

MORE SCIENTIFIC:

A MATHEMATICAL FORMULA THAT REPRESENTS A COMPLEX SYSTEM

AND CAN BE USED TO PREDICT ITS PROPERTIES

"All models are wrong, but some are useful." George E. P. Box, British statistician

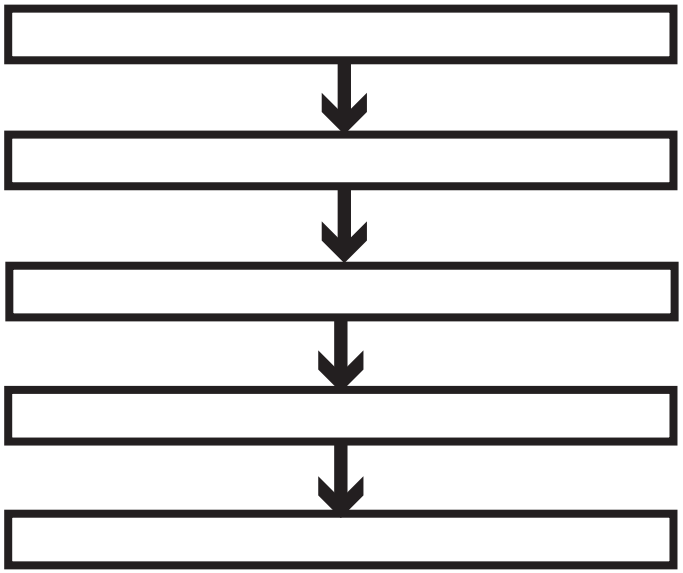
A goal of this course is to be able to compose networking patterns to design or understand network architectures.

Composition is challenging, and requires formality to be credible.

LAYERS OF CLASSIC INTERNET OR OSI REFERENCE MODEL

there is a fixed number of layers

each layer has a distinct and indispensable function

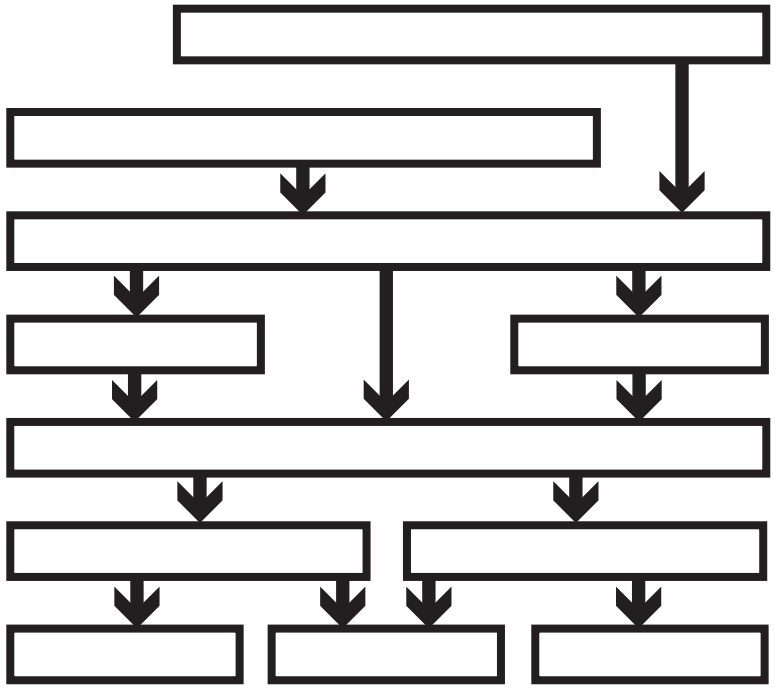


A COMPOSITIONAL MODEL OF NETWORKING

a network is a module

each network/module is a microcosm of networking, containing all the basic functions (state components and algorithms)

because there is a standard module, there is a standard form of composition



in a hierarchy, there can be any number of levels, each with any number of networks

PURPOSES OF A MODEL

DESCRIPTION

- provide well-defined terminology
- enable precise comparison of architectures

DESIGN

- identify the important problems and solution patterns
- reduce complexity through the use of patterns and composition rather than *ad hoc* designs
- generate expanded design spaces
- facilitate the recognition of design principles and structured trade-off spaces
- expand the range and utility of network services

OPPORTUNITIES FOR AUTOMATION

- hierarchical, compositional reasoning about properties
- optimal placement of functions
- code generation and re-use
- automated verification

how are desirable properties defined, and where is the information needed to verify them?

EDUCATION

- help people understand networking more quickly and more deeply

learn patterns and principles, not (just) details

Jennifer Rexford said:

the study of networking is cool

*** young, relatively immature field**

- great if you like to make order out of chaos

- tremendous intellectual progress is still needed

- YOU can help decide what networking really is

*** defining the problem is a big part of the challenge**

- recognizing a need, formulating a well-defined problem

- . . . is at least as important as solving the problem

This is not the only thing about networking she said!

But if you like this part of what she said,

***Patterns in Network Architecture* is for you.**

HOW THE COURSE WILL BE CONDUCTED

office hours: right after the class, or by appointment

PRE-CLASS ASSIGNMENTS (20% of grade)

- read papers
- turn in a simple assignment

answer a question or two about the reading, model something small in Alloy —this is just to get you thinking

CLASSROOM TIME (40% of grade)

- some lecturing, so you don't have to read everything
- discussion of the papers
- discussion of Alloy models

not for general knowledge, but for the purposes of this course!

FINAL RESEARCH PROJECT (40% of grade)

- proposal, presentation, and report
- we will discuss possible topics as we go along

in particular, the first four weeks are very dense

WARNING

- schedule will change, I don't know how much time material will take

FINAL RESEARCH PROJECT

WHAT

- open-ended research problem
- possible topics will be mentioned in class

WHO

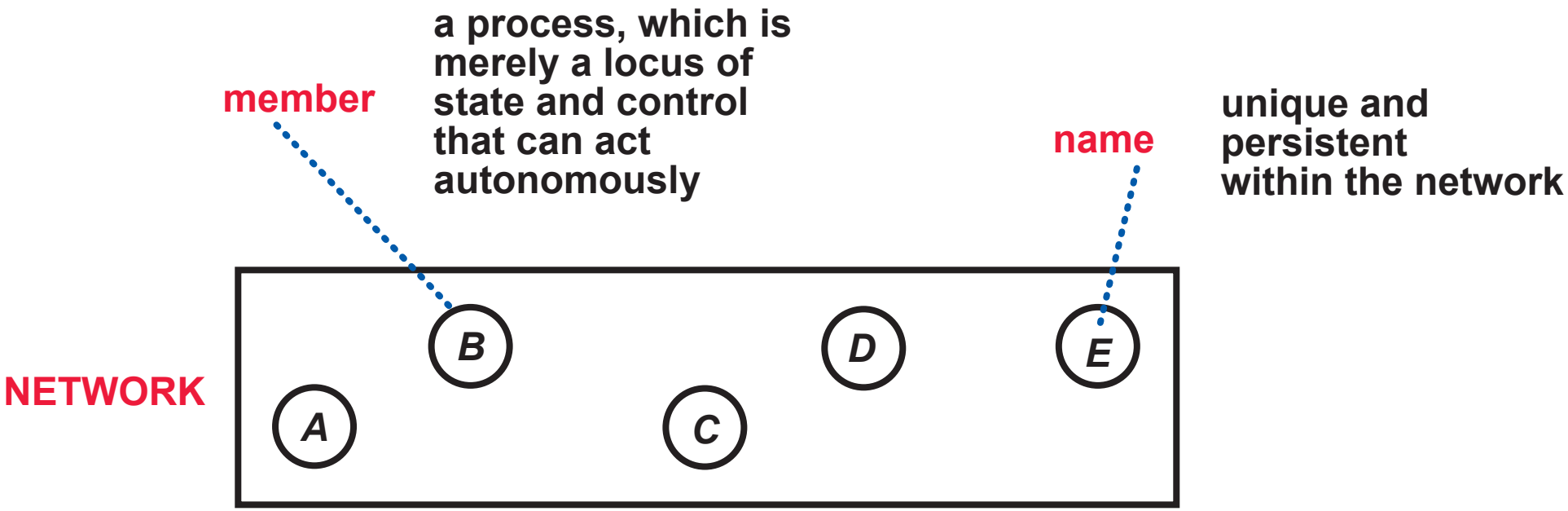
- you can work alone or in small teams
- you can overlap it with another project, with permission

e.g., undergraduate independent work, graduate student research project

WHEN

- short proposal due 28 March
- interim presentation 2 May (last class)
- 6-page final report due 16 May (Dean's Date)

A NETWORK MODEL: MEMBERS AND NAMES



IP interfaces of machines . . .

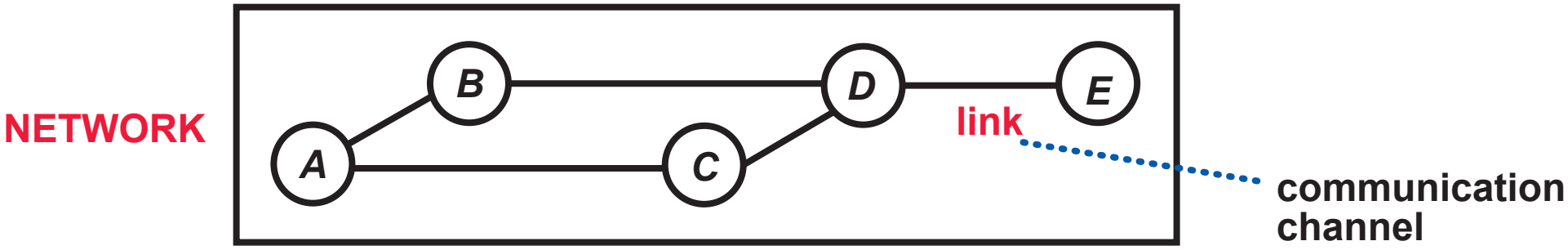
. . . are members of an IP network

IP addresses are their names

there are many other members and names, at different levels

- **domain names (Web servers)**
- **email addresses (mail clients)**
- **MAC addresses (Ethernet interfaces)**

A NETWORK MODEL: ROUTING AND FORWARDING



forwarding protocol . . .
. . . enables members to send messages to one another, using the links

forwarding tables . . .
. . . are the network state that control the forwarding protocol

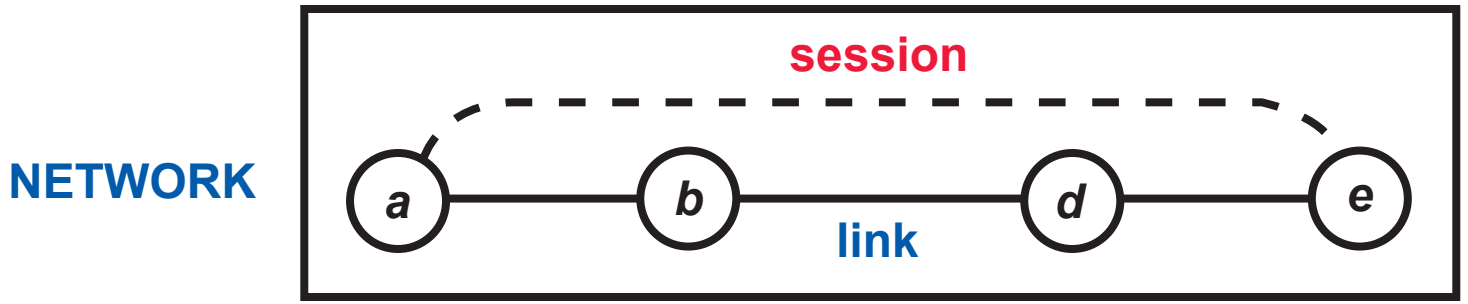
routing algorithm . . .
. . . maintains the forwarding tables as links change over time

there are many networks, with different topologies, at different levels

- **the IP layer is a general graph**
- **an Ethernet LAN is a spanning tree**
- **an application layer may be fully connected (direct virtual link between every pair of application agents)**

A NETWORK MODEL: END-TO-END SERVICES

session protocol implements an end-to-end communication service, on top of the basic message delivery provided by the forwarding protocol



TCP is a session protocol . . .

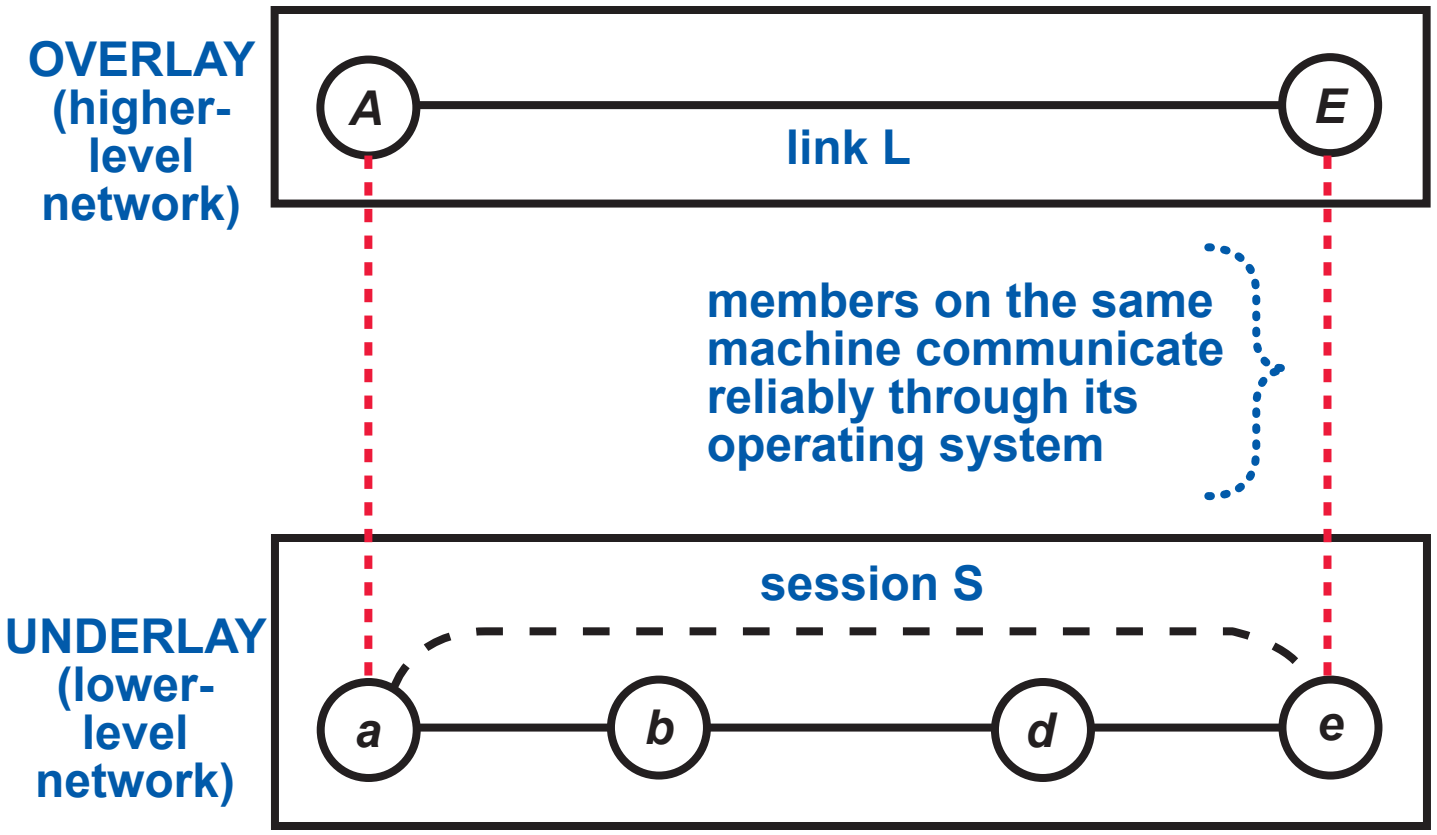
. . . providing reliable, FIFO, duplicate-free message delivery

there are many other session protocols providing different services, for different purposes, at different levels

- SIP (voice and multimedia)
- HTTP (Web)
- IPsec (encryption)

A NETWORK MODEL: THE "USES" HIERARCHY OR VERTICAL COMPOSITION

when an overlay uses an underlay,
a link in the overlay is implemented
by a session in the underlay



in a directory, . . .

E is attached to e,
or . . .

e is the location of
E.

at the endpoints,...

L uses S, or . . .

S implements L.

possible setup of this link/session:

1 *A* sends request to *a*
2 *a* looks up location of *E*, finds *e*
3 *a* sends request to *e*
4 *e* sends request to *E*

A NETWORK MODEL: MAJOR PARTS

PROTOCOLS

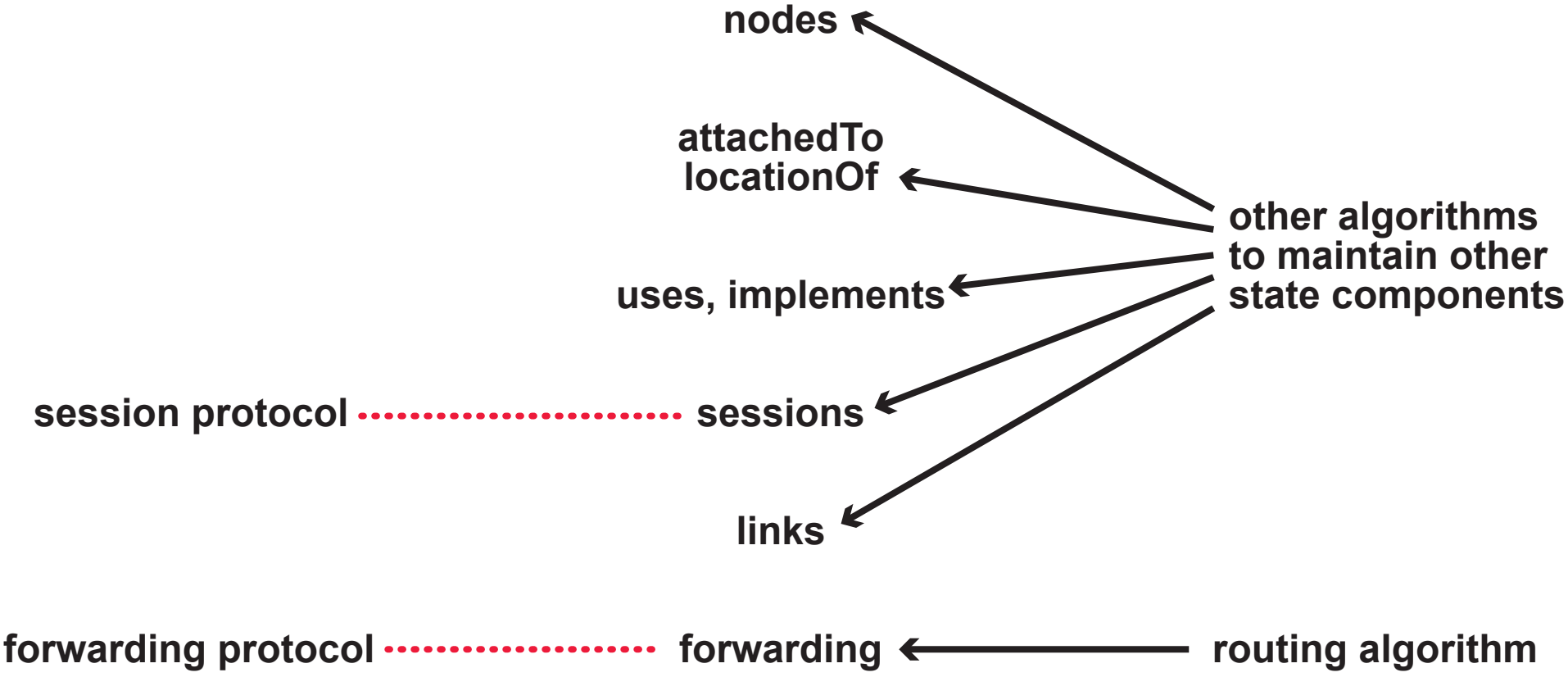
or, the "data plane"

STATE COMPONENTS

can be centralized or distributed across the members in any way

ALGORITHMS

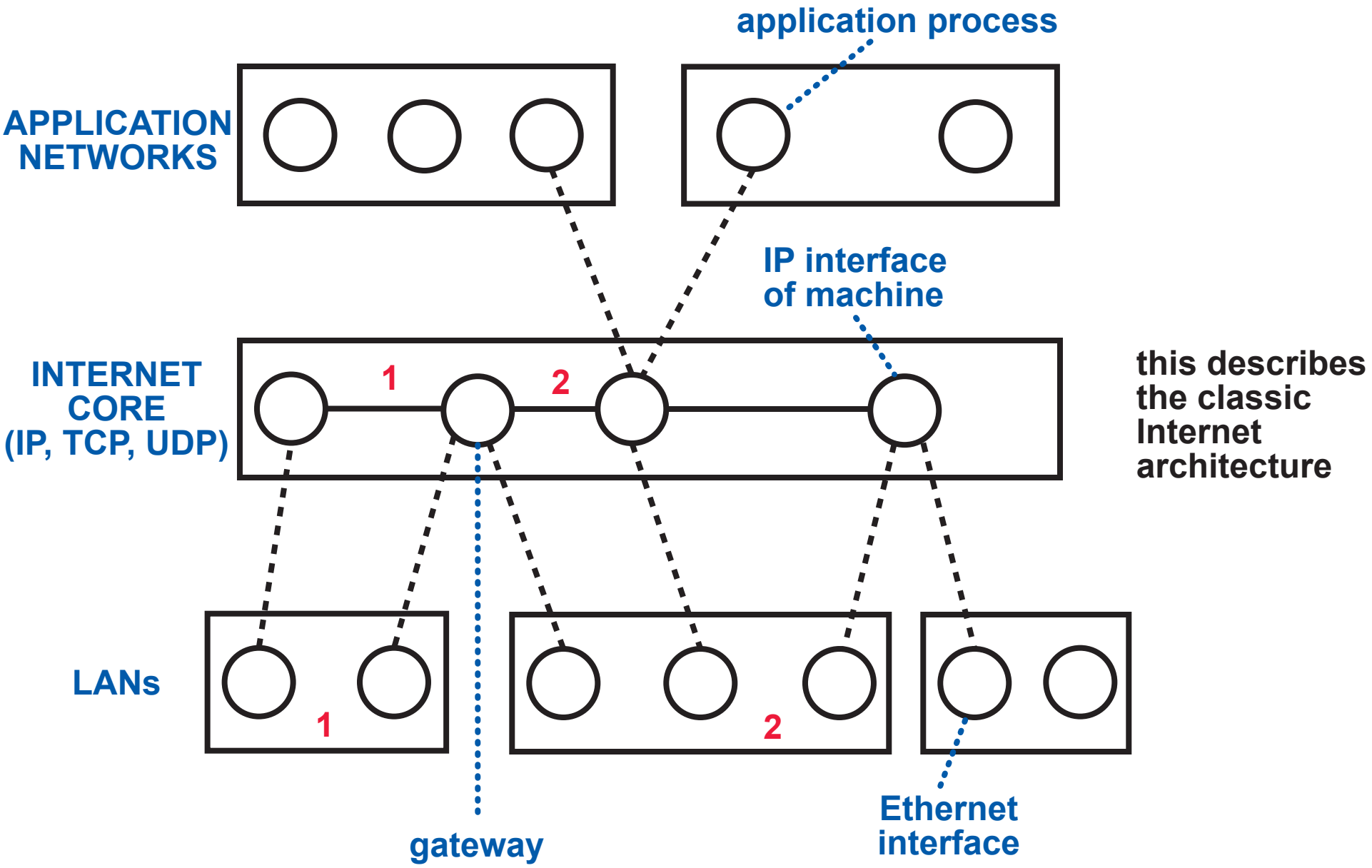
or, the "control plane"



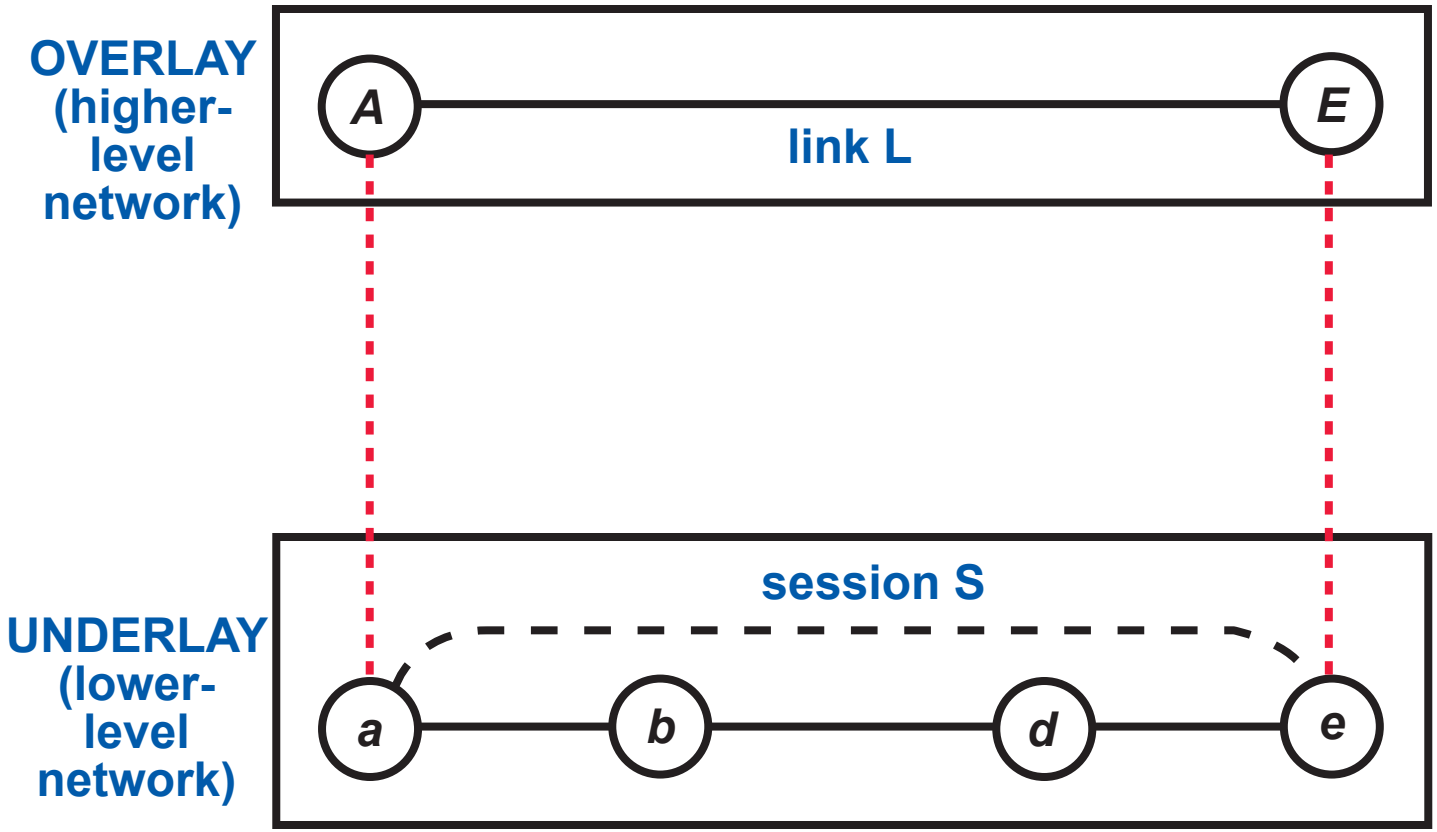
A NETWORK MODEL: SCOPE AND LEVEL

networks are arranged in a "uses" hierarchy, which defines levels

the scope of a network is the set or class of processes that could be member nodes



A LOOK AHEAD



if L and S are being set up dynamically, . . .

then the required parts of the session header are . . .

source: *a*

destination: *e*

sessionIdentifier: *unique at a*

network: *overlay*

HOW TO READ A PAPER

it's a little different for this course!

WHAT WE CARE ABOUT

- what generalizable network problems are being solved?
- what patterns are being used to solve them?
- which networks are being composed? what are the correspondences between parts where the composition takes place?
- if the model tells you that some state or algorithm is necessary, where is it?
- how would you specify correctness?

WHAT WE MIGHT SKIP

- implementation details
- performance evaluation

the model really helps me read papers,
I hope it helps you too

LIGHTWEIGHT MODELING

DEFINITION

- constructing a very abstract model of the core concepts of a system
- using an analysis tool based on exhaustive enumeration to explore its properties

WHY IS IT "LIGHTWEIGHT"?

- because the model is very abstract in comparison to a real implementation, and focuses only on core concepts, it is small and can be constructed quickly
- because the analysis tool is "push-button", it yields results with little effort

in contrast,
theorem proving is not "push-button"

WHAT IS ITS VALUE?

- it is a design tool that reveals conceptual errors early

decades of research on software engineering proves that the cost of fixing a bug rises exponentially with the delay in its discovery

- it is a documentation tool that provides complete, consistent, and unambiguous information to implementors and users
- it is easy (at least to get started) and surprising (you get the result of scenarios you would *never* expect)

"If you like surprises, you will love lightweight modeling."
—Pamela Zave

- EASY + SURPRISING = FUN

WHY IS LIGHTWEIGHT MODELING EASY, SURPRISING?

EASY + SURPRISING = FUN

PROGRAMMING:

1 write a program

2 think of a test case

3 run the program on the test that you thought of

LIGHTWEIGHT MODELING

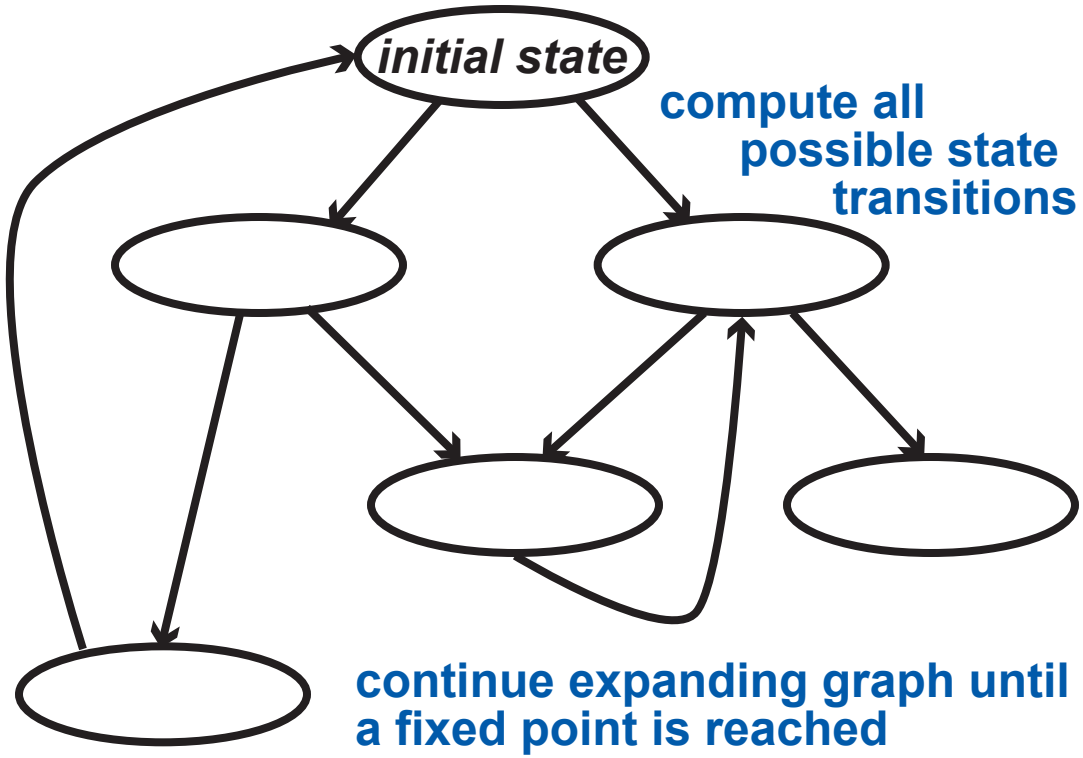
1 write a model (no bigger than a small program)

2 push the "analyze" button

3 get results from *all possible executions* in a particular category, including "tests" you would *never* have thought of!

HOW MODEL CHECKERS DO IT

all data structures have fixed size, so state space is bounded (includes implicit structures such as call stack)



the result is an explicit, finite reachability graph representing all possible states, state transitions, and traces (finite or infinite paths through the graph)

RELATIONAL JOIN

THE KEY TO UNDERSTANDING

RELATIONAL ALGEBRA (AND ALLOY)

RELATIONS

P is of type A	Q is of type A -> B -> C	R is of type C -> D
	A$0 -> B$0 -> C$0	C$0 -> D$0
A$1	A$1 -> B$1 -> C$1	C$1 -> D$1
A$2	A$2 -> B$2 -> C$2	

JOIN EXPRESSION

P . Q . R .....columns on either side of dot must have same type

COMPUTATION OF JOIN

value in "shared column" must match

A$0 -> B$0 -> C$0	C$0 -> D$0	in resulting relation, "shared columns" are removed
A$1 -> B$1 -> C$1	C$1 -> D$1	
A$2 -> B$2 -> C$2		

VALUE OF JOIN EXPRESSION

B$1 -> D$1

result is a relation with any number of tuples, including zero or many

WHAT IS THE HIDDEN CHALLENGE?

It is so easy to write a model, ask the analyzer a question, get an answer . . .

. . . but not so easy to know what any of these means in the real world.

NONDETERMINISM IN MODEL

- environment choice
- implementation freedom
- system failure
- concurrency

STATEMENTS IN MODEL

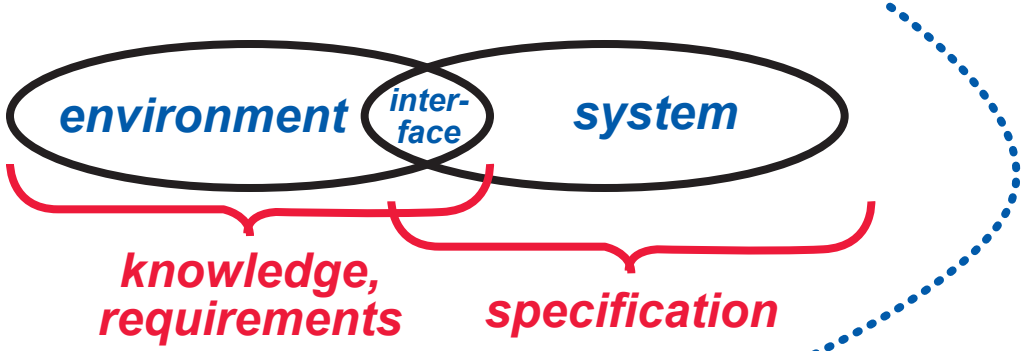
- domain knowledge: description of the environment in which the system will operate (fact or assumption)
- specification: an implementable description of how the hardware/software system should behave
- requirement: a description of how the environment should behave when the system is implemented and deployed
- sanity check: intended to be redundant

ANALYSIS QUESTIONS

- Is the model consistent (can be realized) ?
- Are the knowledge and requirements correct ("validation") ?

sanity checks help

- Is the specification correct ("verification") ?



knowledge & specification => requirements

[Jackson & Zave 95]

NET1SPEC AND NET1IMPL:

WHAT ASSUMPTIONS ARE WE MAKING

IF WE USE THEM?

NET1SPEC AND NET1IMPL:

WHAT ASSUMPTIONS ARE WE MAKING

IF WE USE THEM?

INFORMAL, AT LEAST AT THIS POINT

- ***all*** **nodes of the network conform to the state distribution and functional semantics described in the comments of net1impl**
- **the nodes and links work fast enough so that overall behavior is useful**

  is a SneakerNet tolerable?

FORMALIZED FOR ANALYSIS

- **among the network properties, which is assumed, which are we trying to prove?**

ARE WE ASSUMING THIS ONE?

- **all the links are working (not failed)**

NET1SPEC AND NET1IMPL:

WHAT ASSUMPTIONS ARE WE MAKING

IF WE USE THEM?

ARE WE ASSUMING THIS ONE?

- all the links are working (not failed)

BASIC (LTL) TEMPORAL PROPERTIES FALL INTO ONE OF TWO CATEGORIES:

SAFETY (something bad does not happen)

negative reach is a safety property

it does NOT depend on the assumption, because if links do not work, the network is still safe

this can all be formalized (although not in Alloy), . . .

. . . but it will take a lot of formal machinery to do it . . .

PROGRESS (something good does happen)

positive reach is a progress property

it DOES depend on the assumption, because if links do not work, good things will not happen

. . . and it will distract us from the all-important network state, . . .

. . . so better to keep it in our heads (and documentation)