Reservation-based NoC timing models for large-scale architectural simulation

Javier Navaridas, Behram Khan,
Salman Khan, Paolo Faraboschi, Mikel Luján

Introduction

- Existing electronic miniaturization technologies allow to integrate several processing cores into a single chip
- General purpose processors provide up to 16 cores
- Many-core processors such as Tilera provide up to 64 cores
- Designing 1000-core processors is a current hot topic
  - Rigel [Kelm et al], ATAC [Kurian et al], TERAFLUX [Portero et al]

Kelm et al. *"Rigel: an architecture and scalable programming interface for a 1000-core accelerator"*

Kurian et al. *"ATAC: a 1000-core cache- coherent processor with on chip optical network"*

A. Portero et al. *"TERAFLUX: Exploiting tera-device computing challenges"*

Evaluating large-scale systems

- ⇨ Traditionally the micro-architecture community has disregarded on-chip communications when evaluating processor designs

- ⇨ With the advent of such large-scale processors, NoC behaviour needs to be taken into consideration

- ⇨ Evaluate such large-scale systems requires a considerable amount of compute power

- ⇨ NoC simulation has to be included in a lightweight manner usually in the form of a timing model

Modelling the NoC for Evaluation

- Full-system simulation
  - Full computational model of the NoC
  - Very high accuracy
  - Expensive in terms of compute power
- Network agnostic timing models
  - Network functionality is not considered
  - Very low accuracy
  - NoC modelling barely affects simulation speed

Modelling the NoC for Evaluation

- Statistical timing models [Papamichael et al]
  - Estimate packet latency from an external analysis of the traffic
    - Traffic analysis may be done concurrently or off-line
  - Improves accuracy without exacerbating compute requirements when compared with network-agnostic models
  - Several limitations
    - Latency distributions are case-specific
    - Latency figures are difficult to estimate for variable traffic patterns
    - Require tracking network load

Papamichael et al. *“FIST: A fast, lightweight, FPGA-friendly packet latency estimator for noc modeling in full-system simulations”*

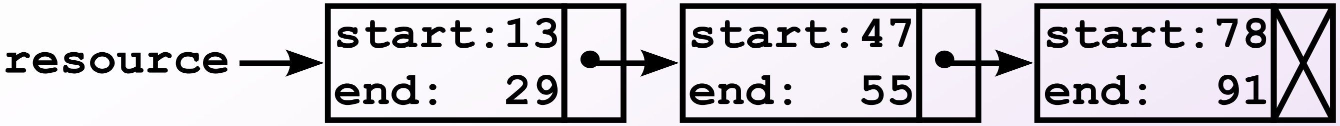
Modelling the NoC for Evaluation

- Reservation-based timing models
  - NoC is modelled in a simple way
    - A collection of resources that need to be reserved to be used
    - If a resource is reserved it can not be used until it is freed
  - Good accuracy
  - Allow fast simulation
  - Avoids the limitations of the statistical models
    - Latency depends on actual state of the network
    - Do not require tracking network load
    - External traffic analysis not needed

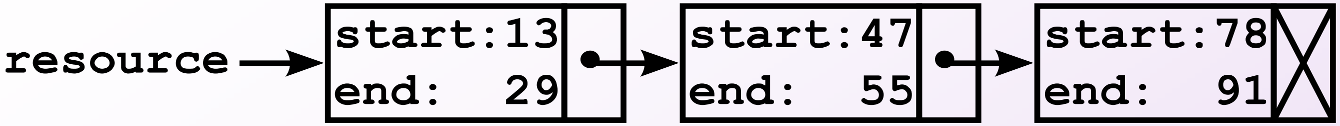
Our Implementation

- Base data-structure
  - Resources are modelled as a sorted linked list which represents the periods in which it is reserved
  - A ‘Reserve’ function to operate over the data-structure
    - Searches for a free period of time that can accommodate a given reservation, reserves the resource and returns the ending timestamp
    - Eliminates outdated reservations and merges existing reservations to keep data structure manageable

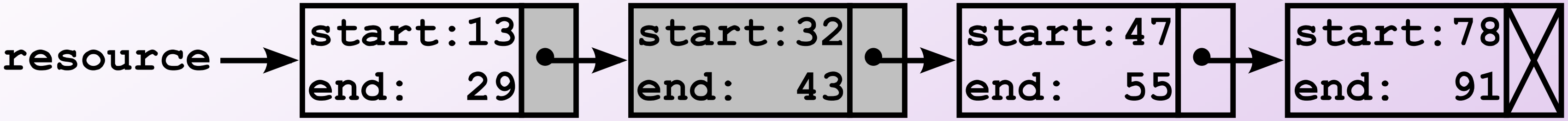
Operation of the Data Structure



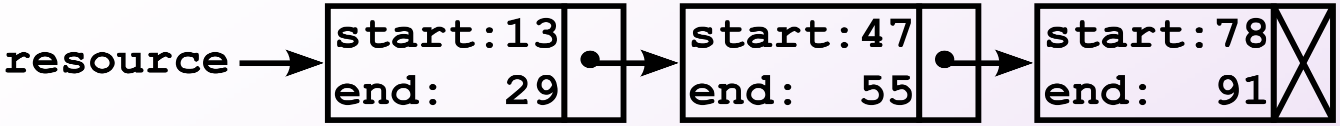
Operation of the Data Structure



timestamp=5 : reserve (32, 11)



Operation of the Data Structure



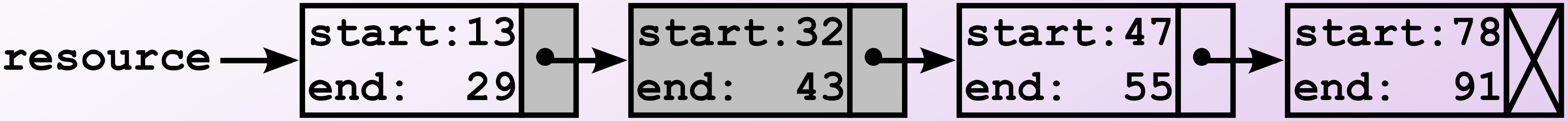
resource

start:13
end: 29

start:47
end: 55

start:78
end: 91

timestamp=5 : reserve (32, 11)



resource

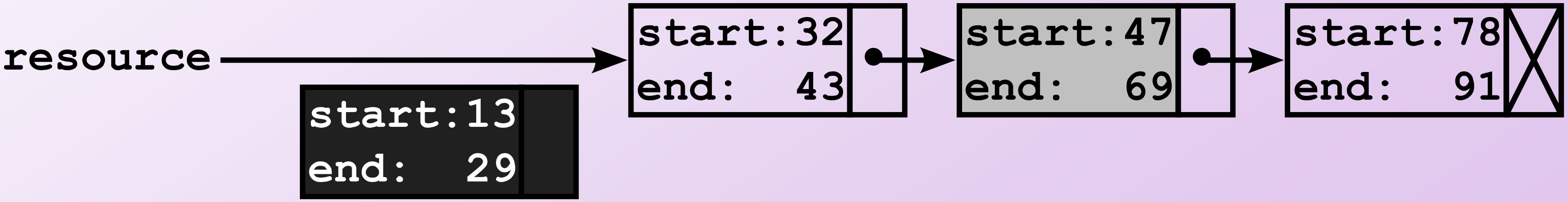
start:13
end: 29

start:32
end: 43

start:47
end: 55

start:78
end: 91

timestamp=40 : reserve (50, 14)



resource

start:32
end: 43

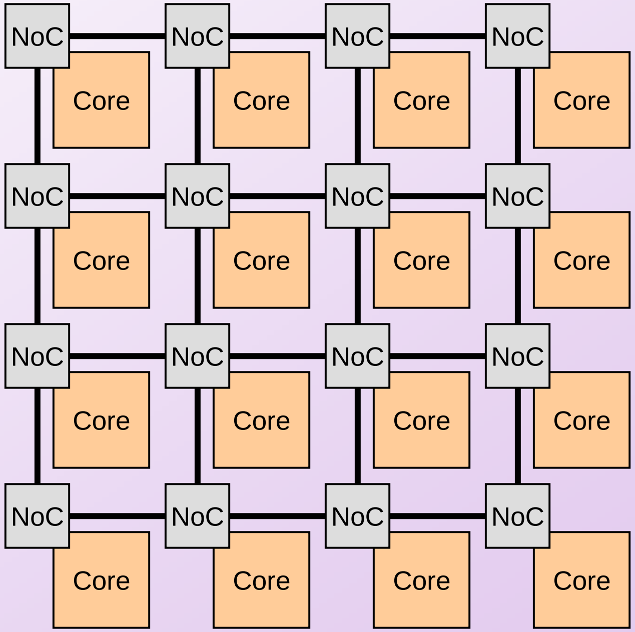
start:47
end: 69

start:78
end: 91

start:13
end: 29

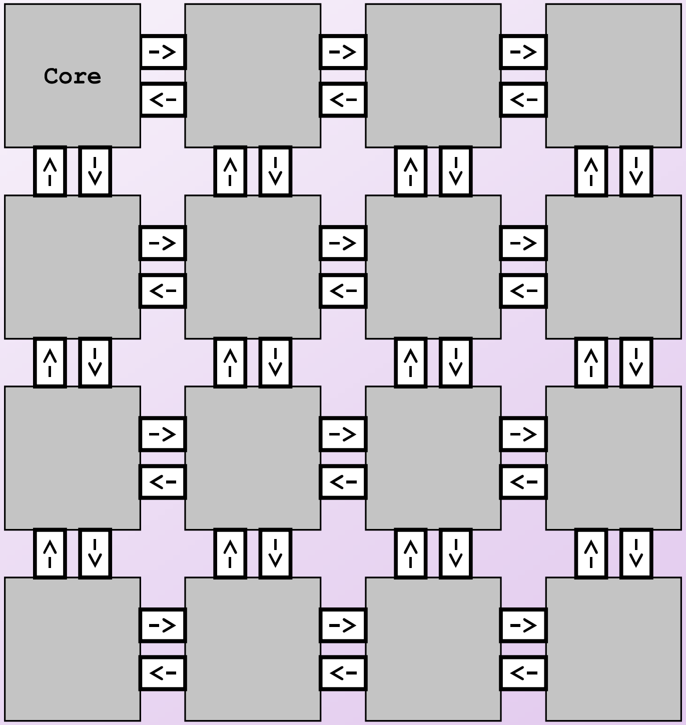
System under Consideration

- ⇨ Mesh topology
- ⇨ XY routing
- ⇨ Cut-through switching
- ⇨ 1 virtual channel



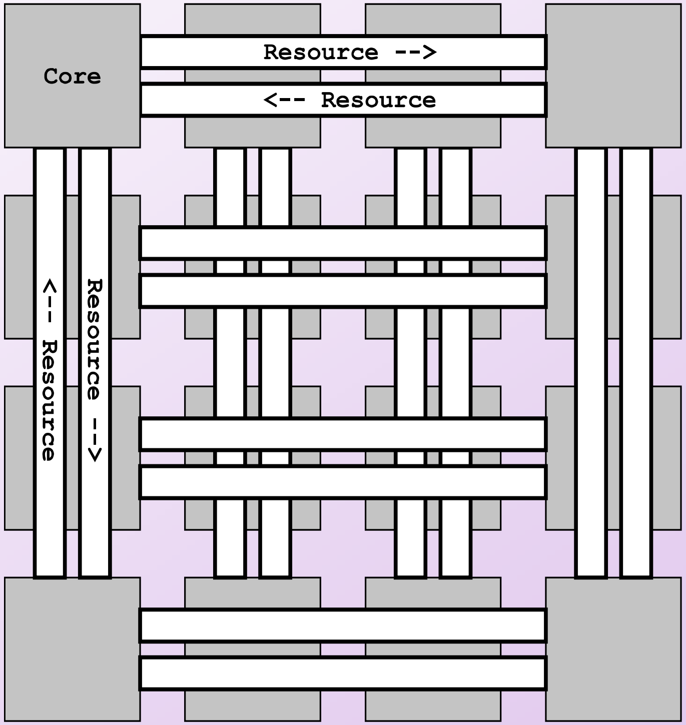
Reservation Models

- NoC modelled at the hop level
  - Each communication link is modelled as a resource
  - Each packet reserves all the required links



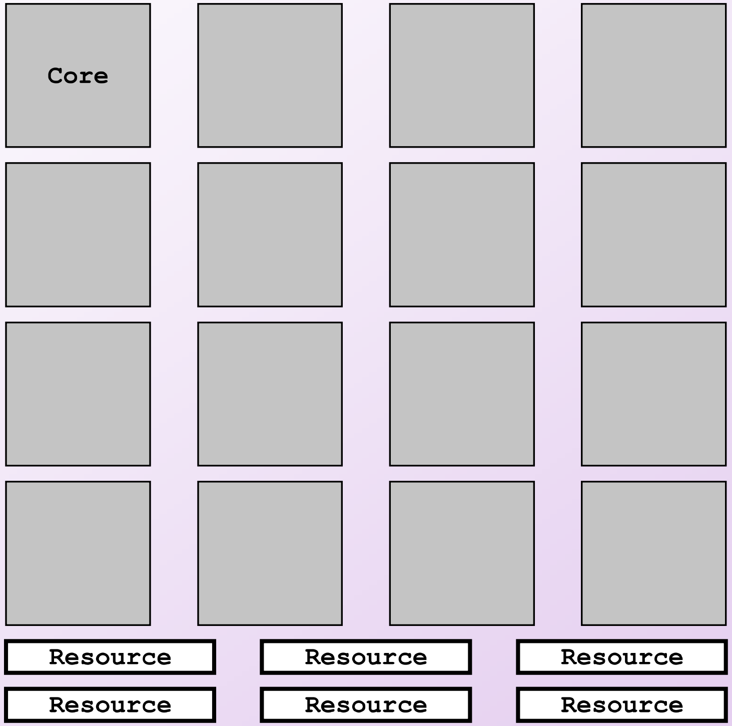
Reservation Models

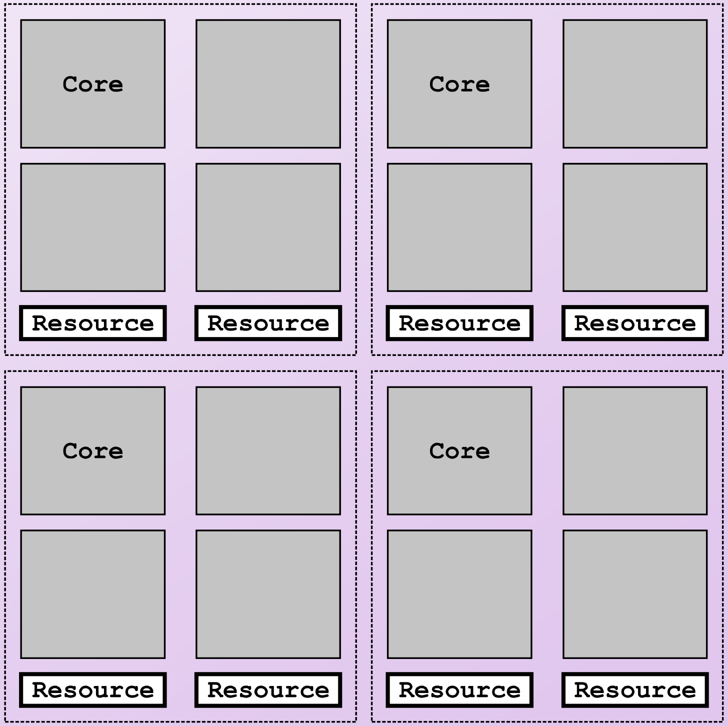
- NoC modelled at the direction level
  - Each row and column of the topology are modelled as a resource per direction (positive/negative)
  - Each packet reserves the required row and column resources



Reservation Models

- Topology-agnostic model
  - Network is modelled as a collection of ‘communication channels’
  - Each packet reserves one of these channels randomly
  - A distributed implementation is also considered





Other Models

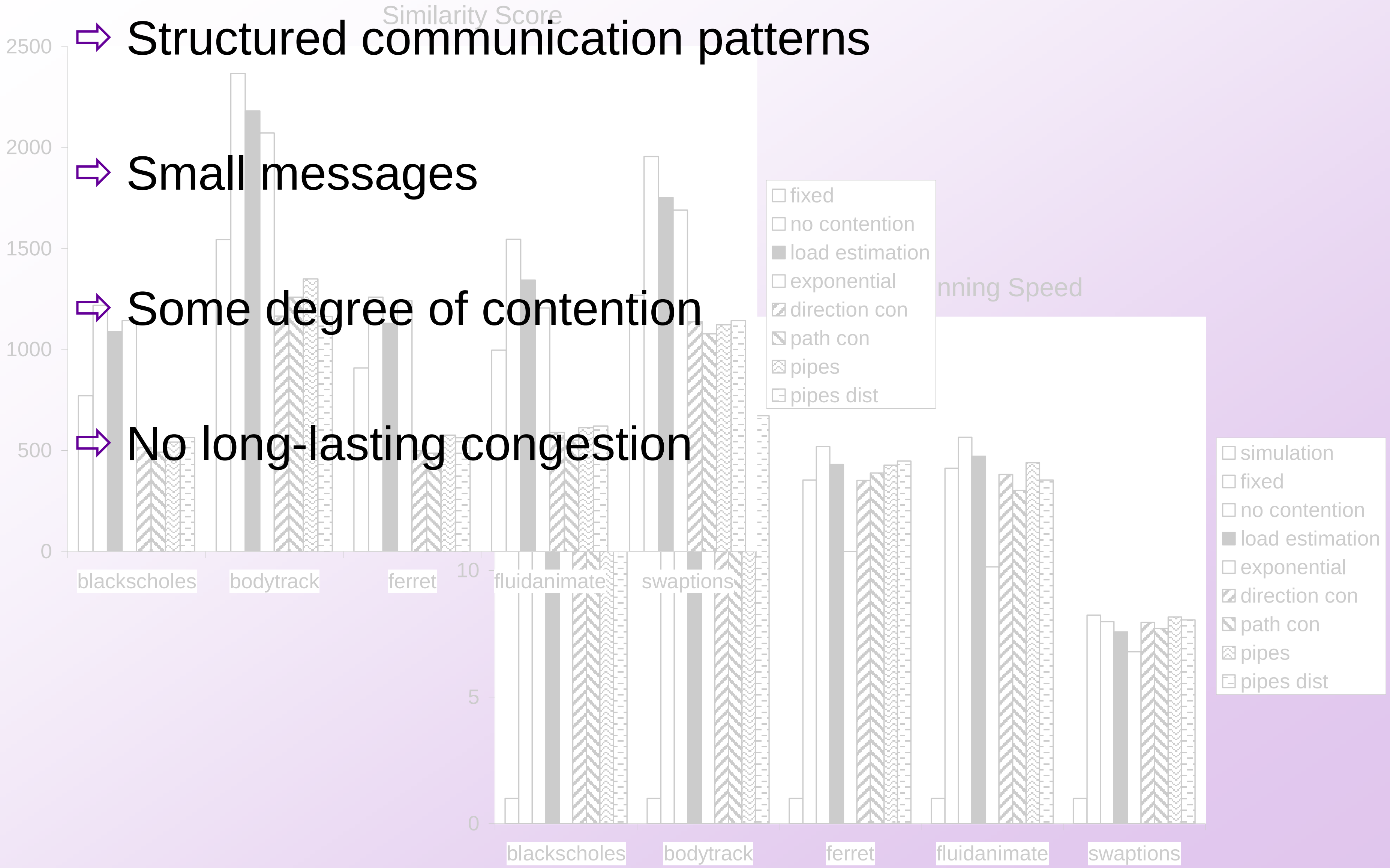
- Network agnostic models
  - Fixed model
    - All network accesses requires the same amount of time
  - No contention model
    - Latency depends only on distance and packet size
- Statistical timing models
  - Load-dependent estimation
    - Tracks the load and models latency in a simple way
      - With low loads latency is barely affected
      - With high loads latency is very high
  - Estimation from off-line simulation
    - Estimate latency from packet distance and average latency

Evaluation

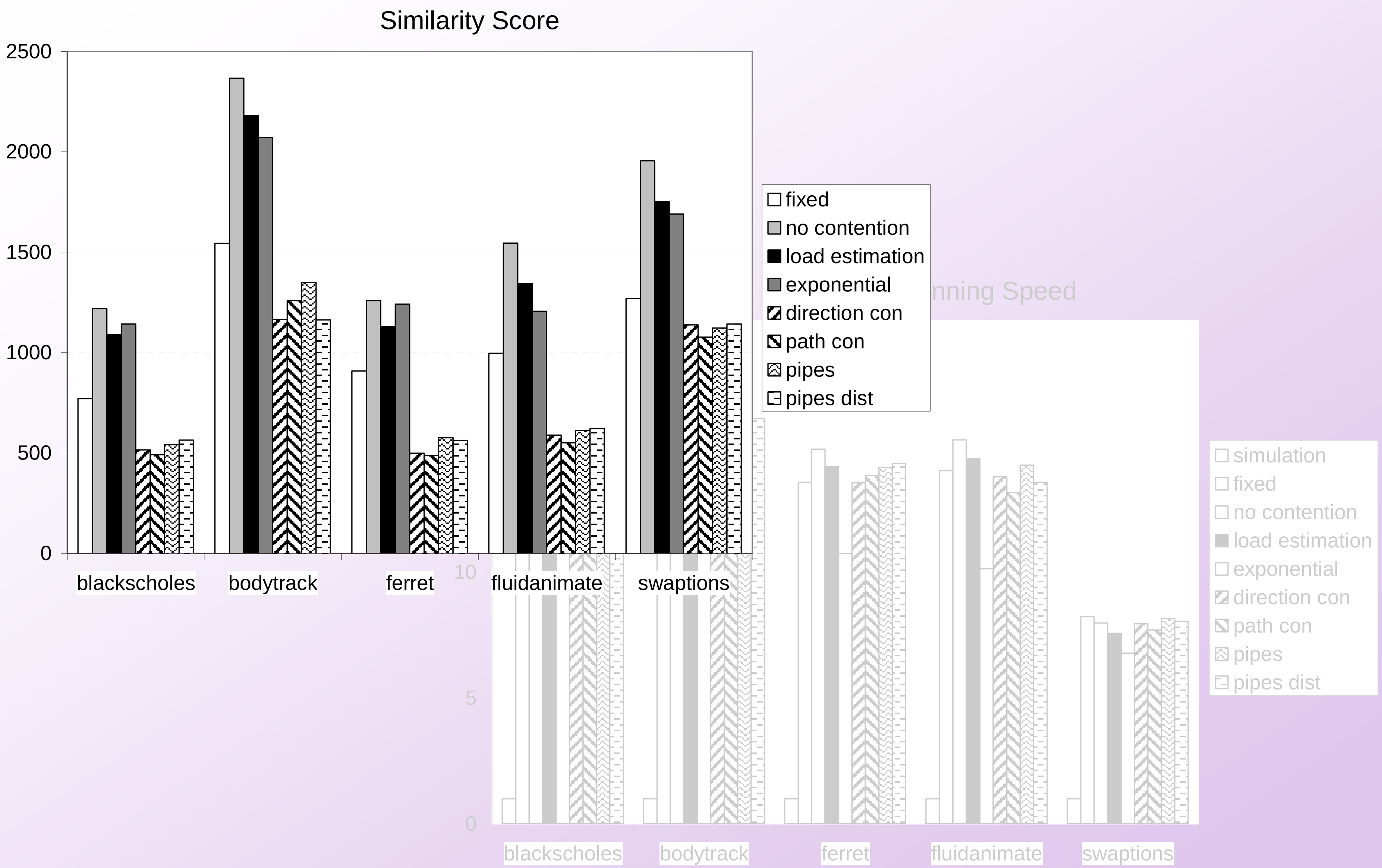
- Models implemented as stand-alone tools
- Trace-driven evaluation
  - PARSEC: Directory-based cache coherency – 32 cores
  - STAMP: Transactional memory – 32 cores
  - NAS: Message passing – 64 cores
  - Cache coherency-like synthetic traffic – 1024 cores
- Figures of merit
  - Accuracy
    - Simulated time to execute the benchmarks
    - Similarity score metric
  - Speed
    - Execution time of the models

PARSEC – 32 cores

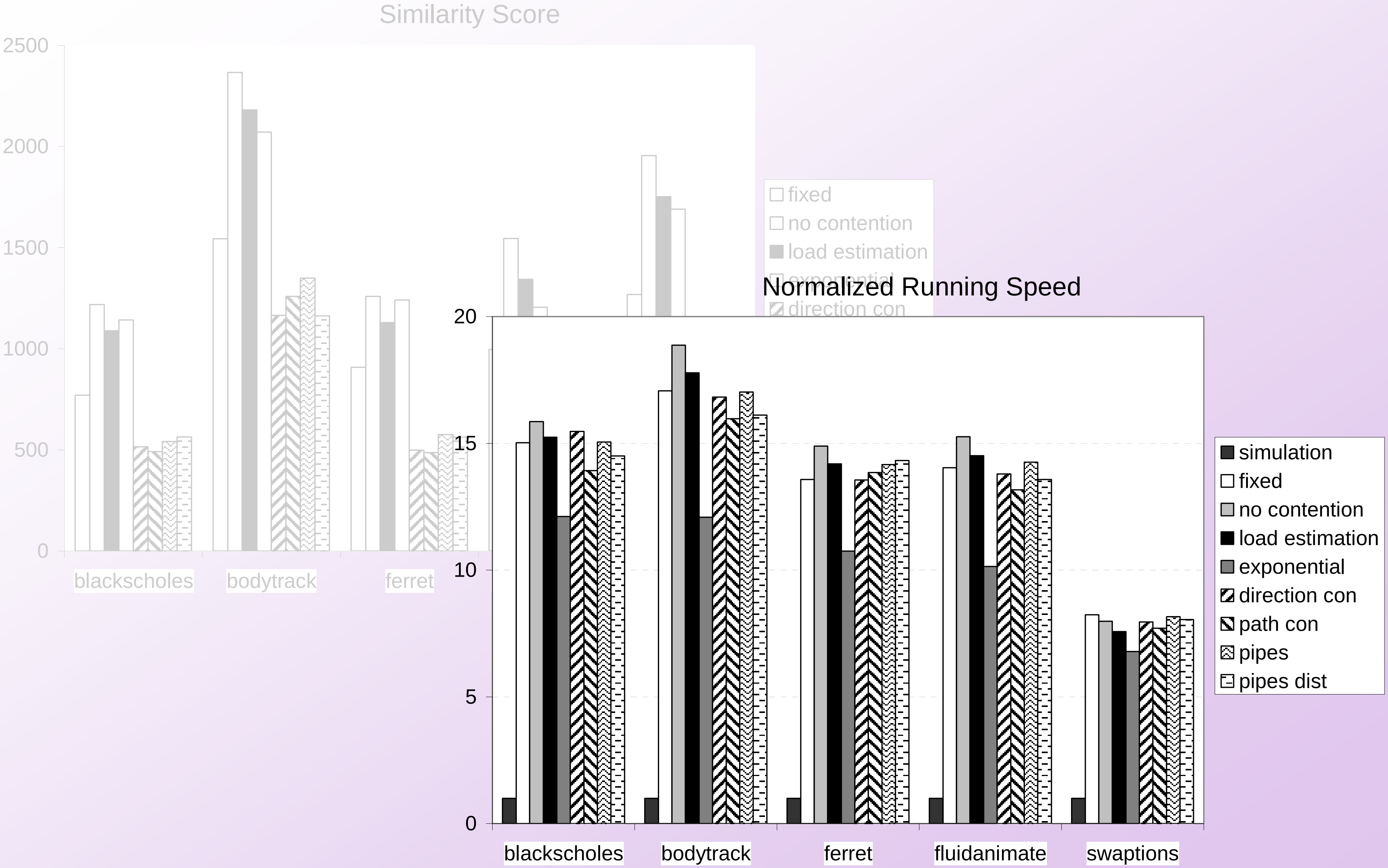
- ⇨ Structured communication patterns
- ⇨ Small messages
- ⇨ Some degree of contention
- ⇨ No long-lasting congestion



PARSEC – 32 cores

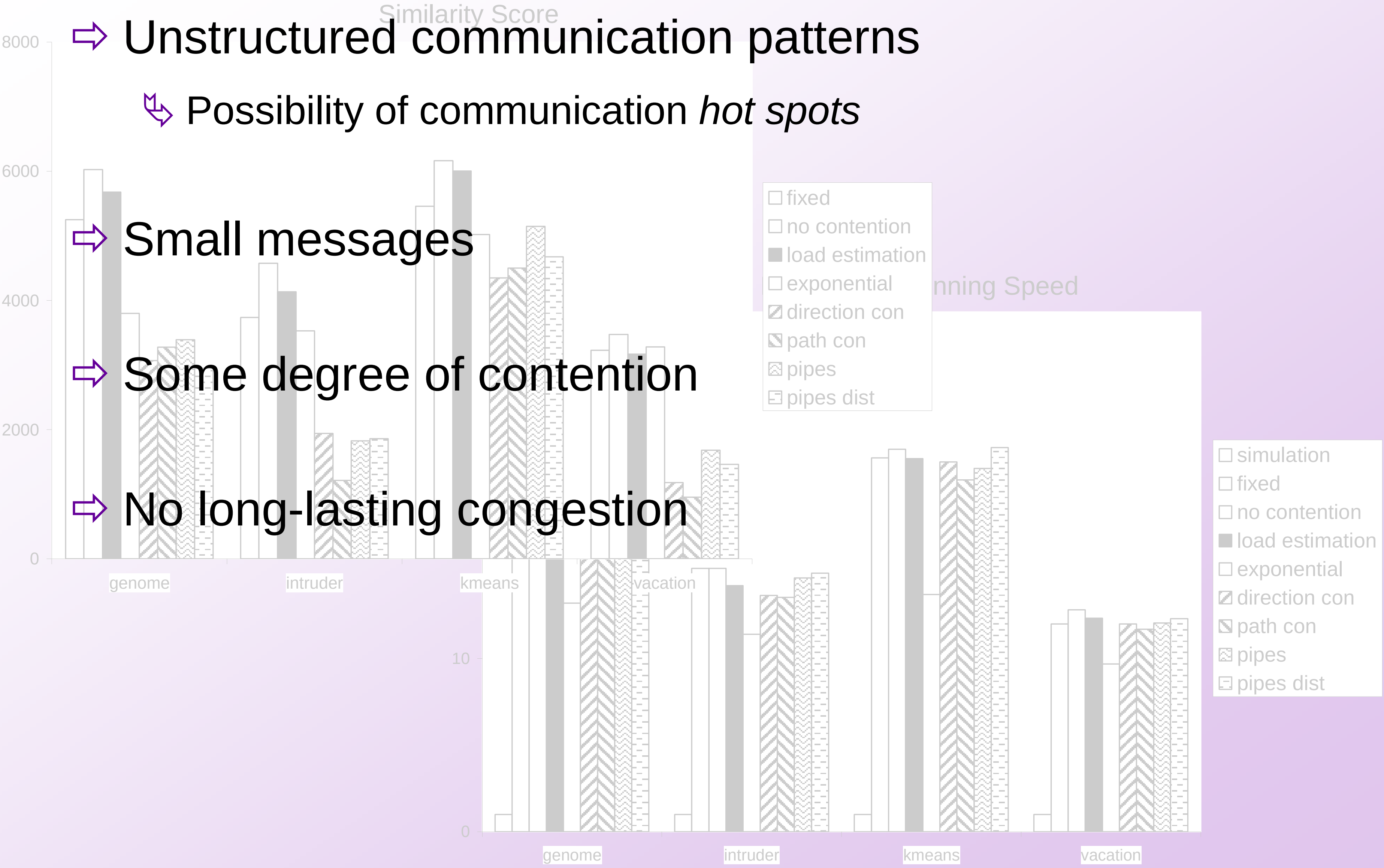


PARSEC – 32 cores

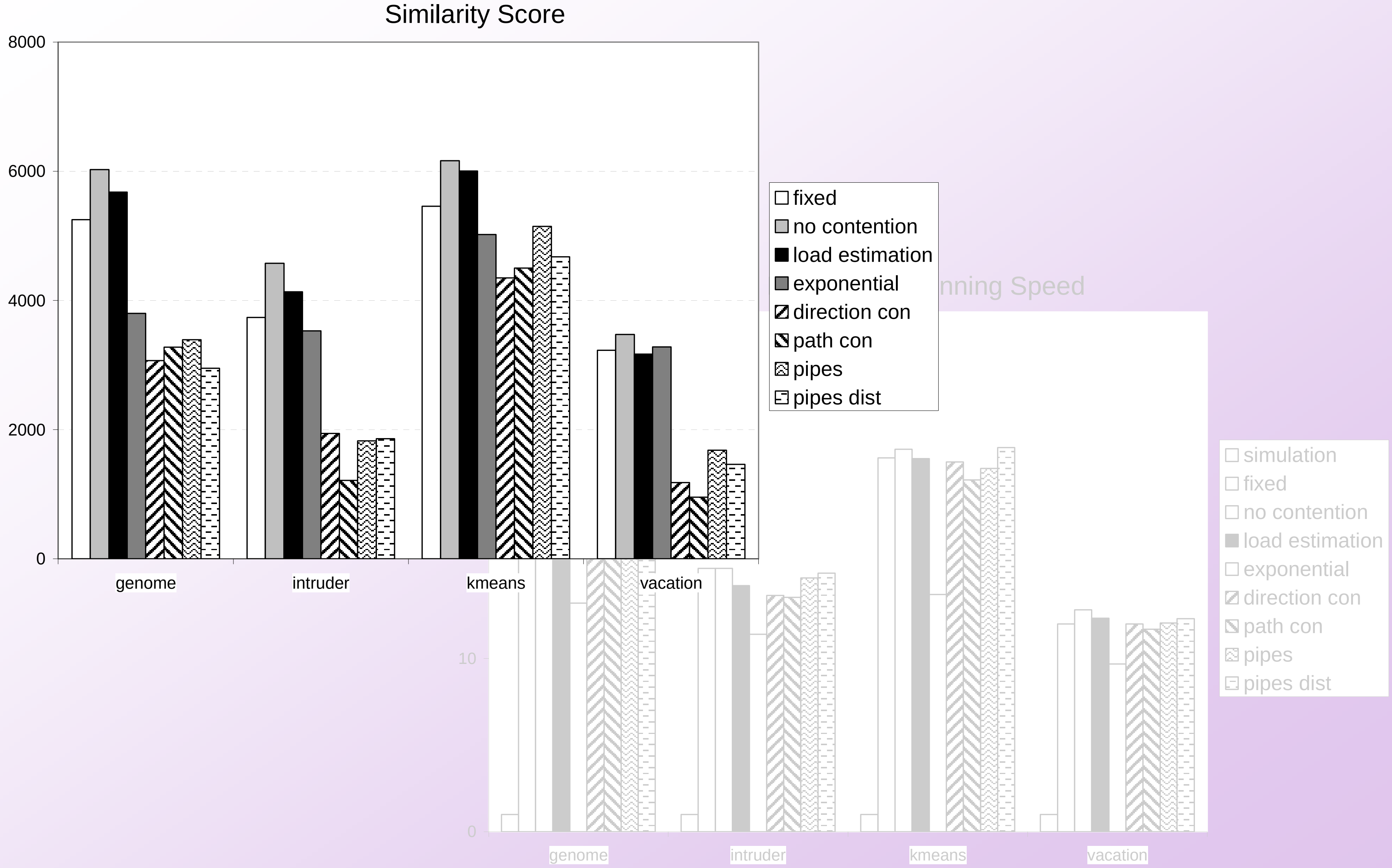


STAMP – 32 cores

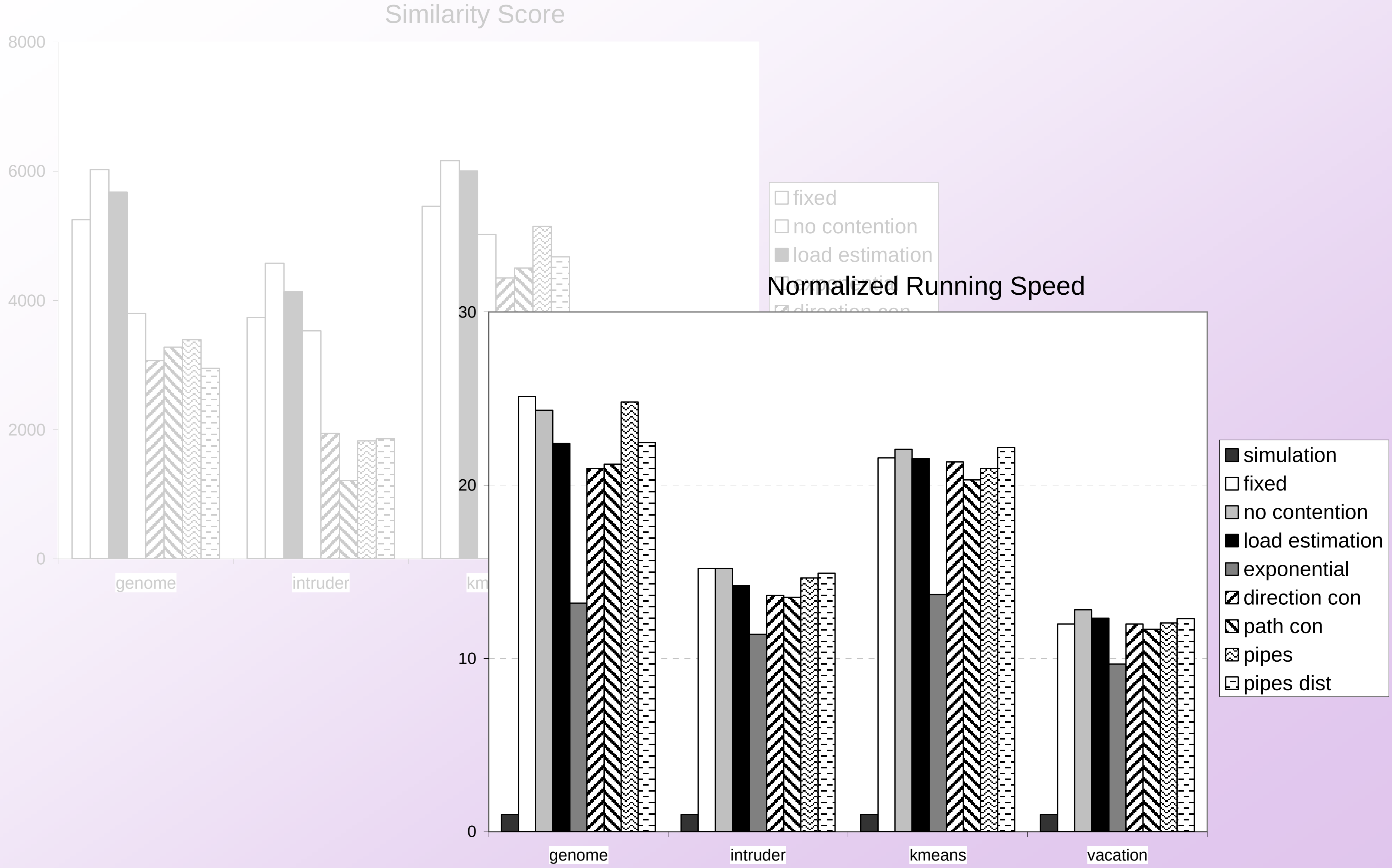
- Unstructured communication patterns
  - Possibility of communication *hot spots*
- Small messages
- Some degree of contention
- No long-lasting congestion



STAMP – 32 cores

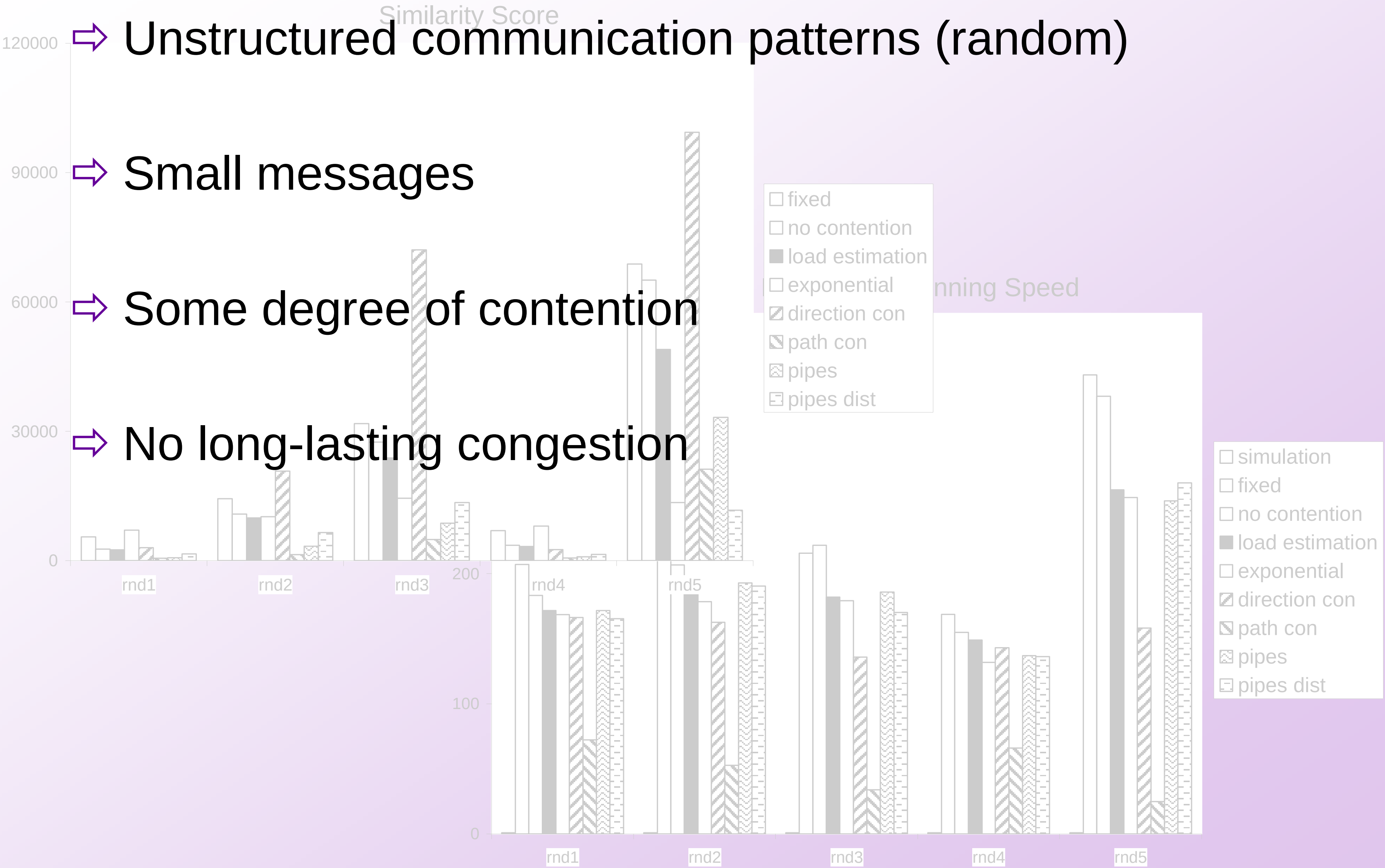


STAMP – 32 cores

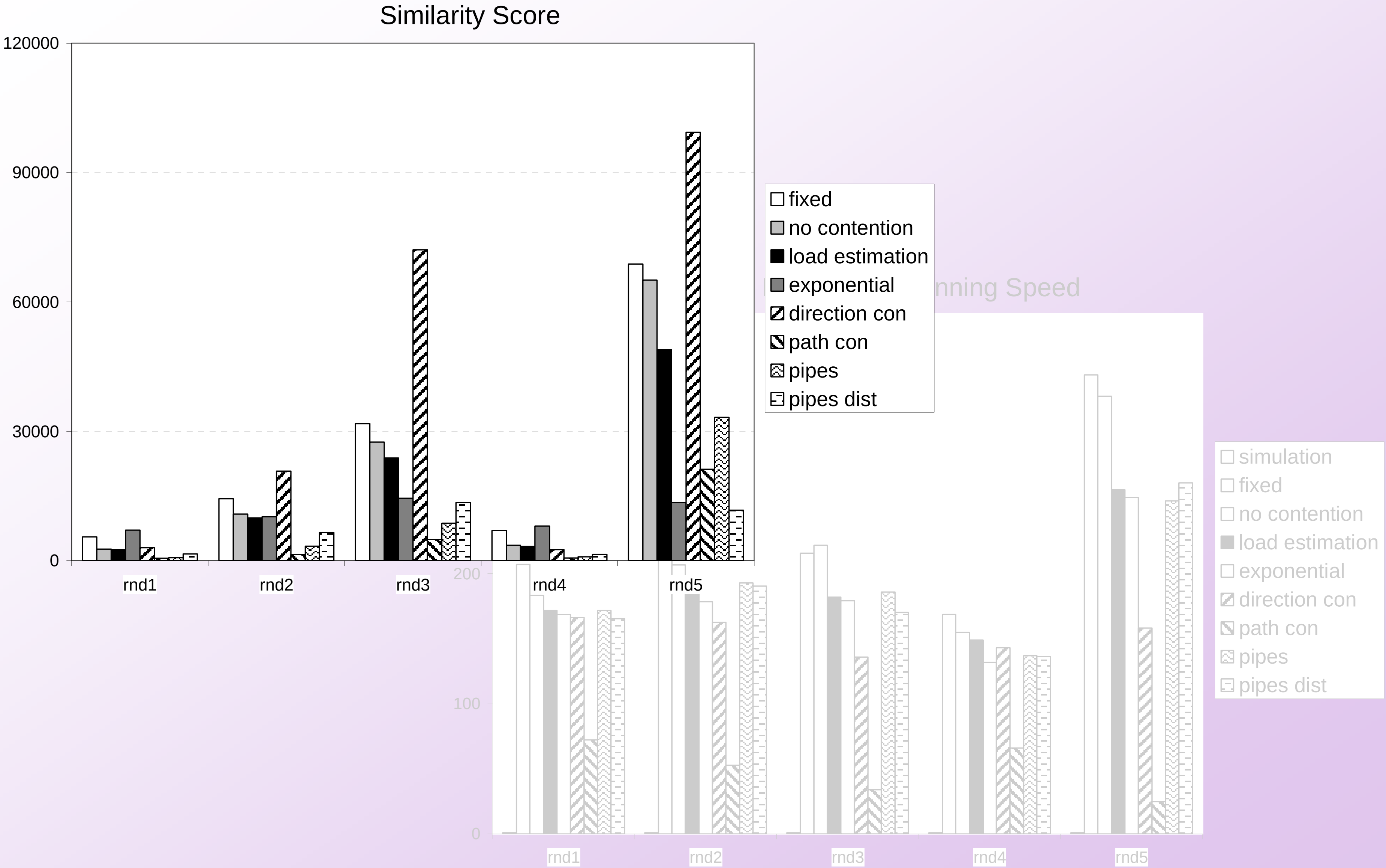


Synthetic – 1024 cores

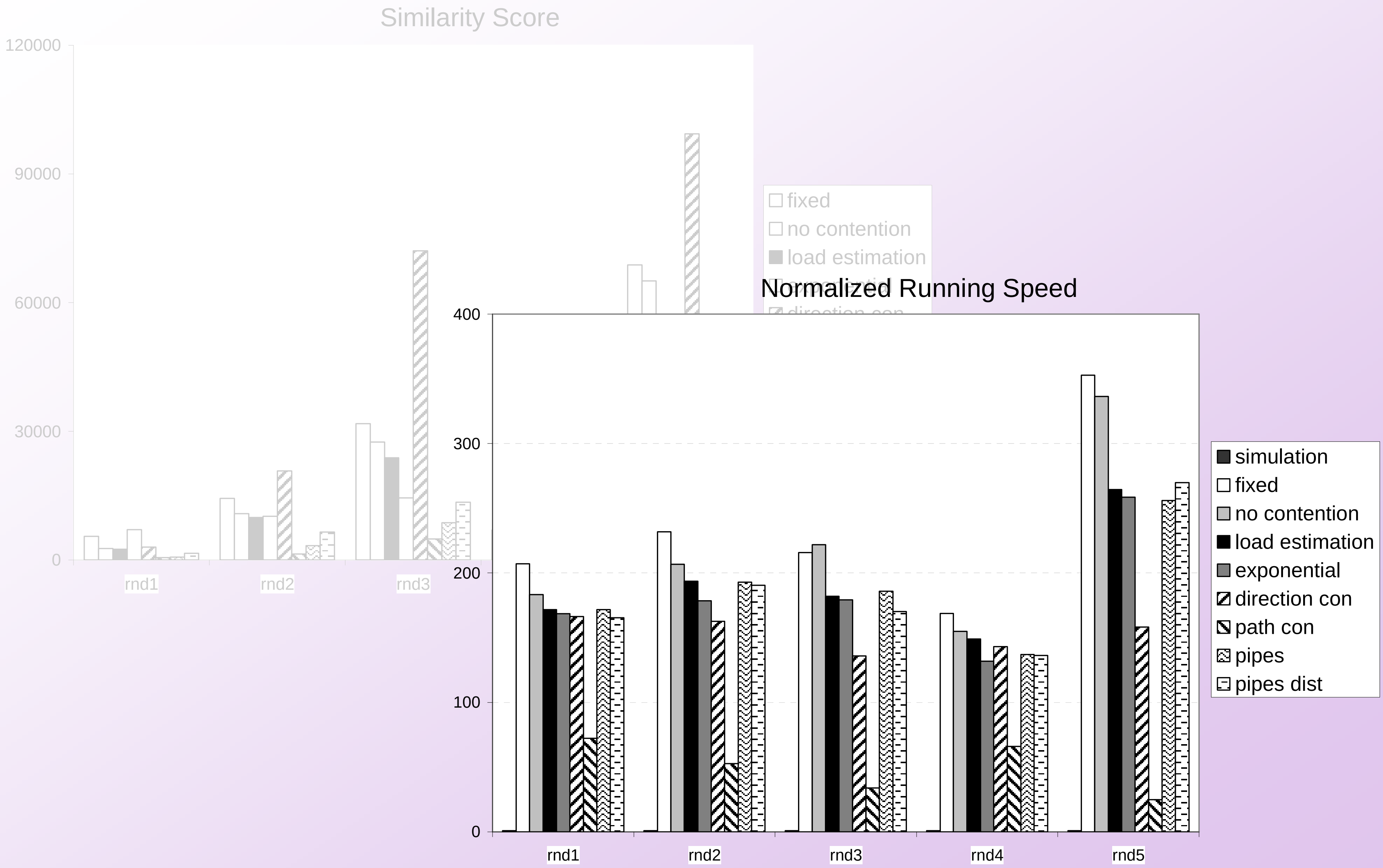
- ⇨ Unstructured communication patterns (random)
- ⇨ Small messages
- ⇨ Some degree of contention
- ⇨ No long-lasting congestion



Synthetic – 1024 cores

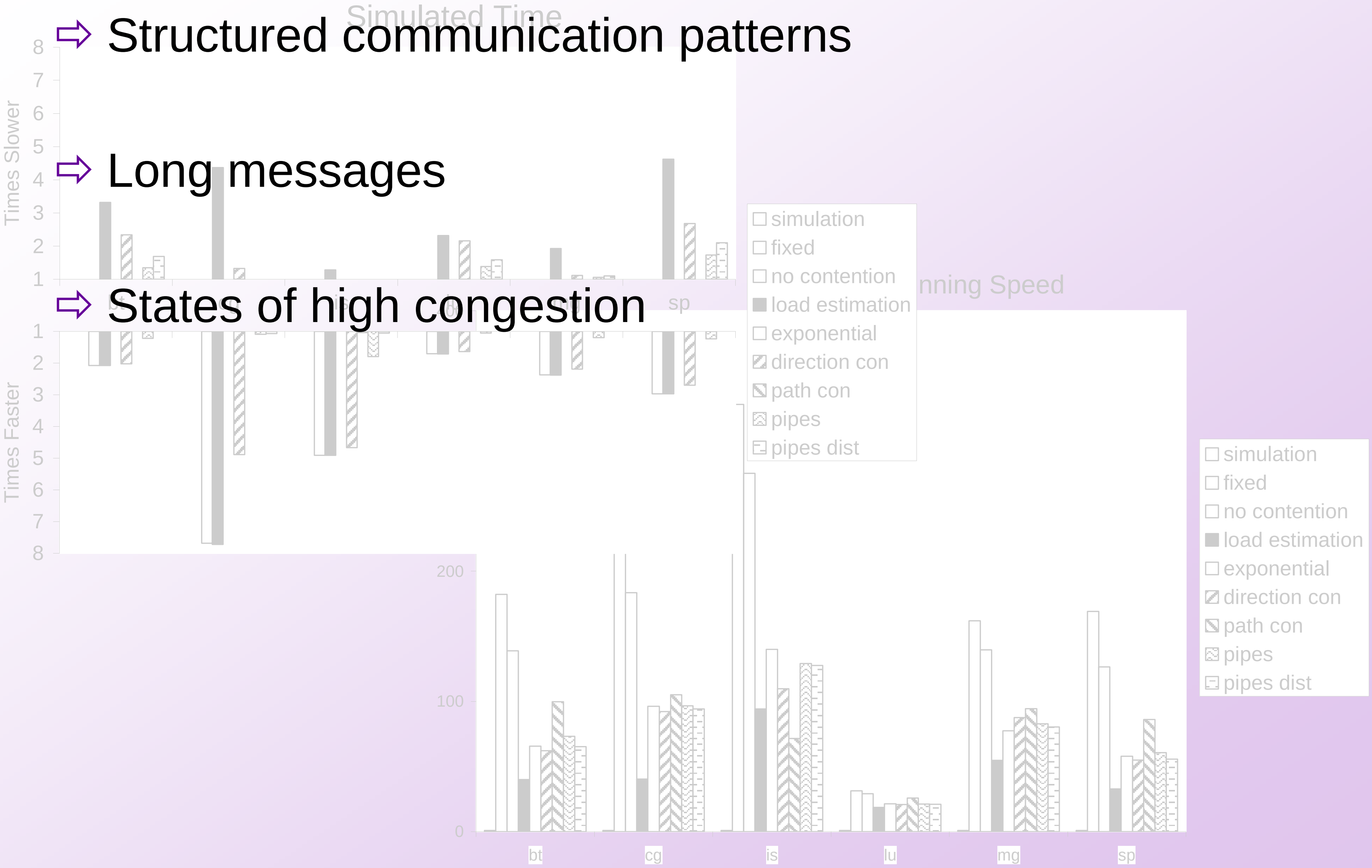


Synthetic – 1024 cores

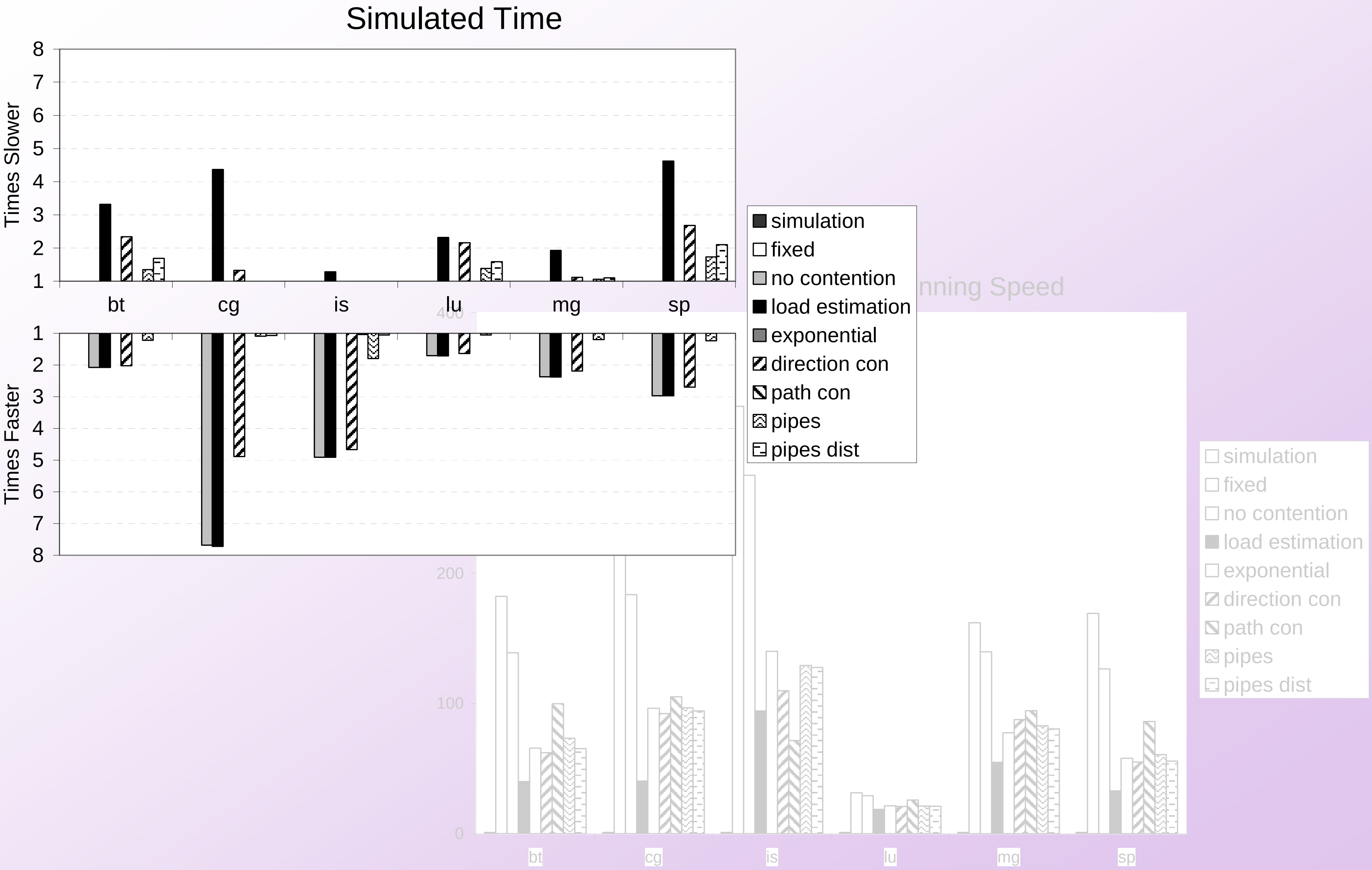


NAS – 64 cores

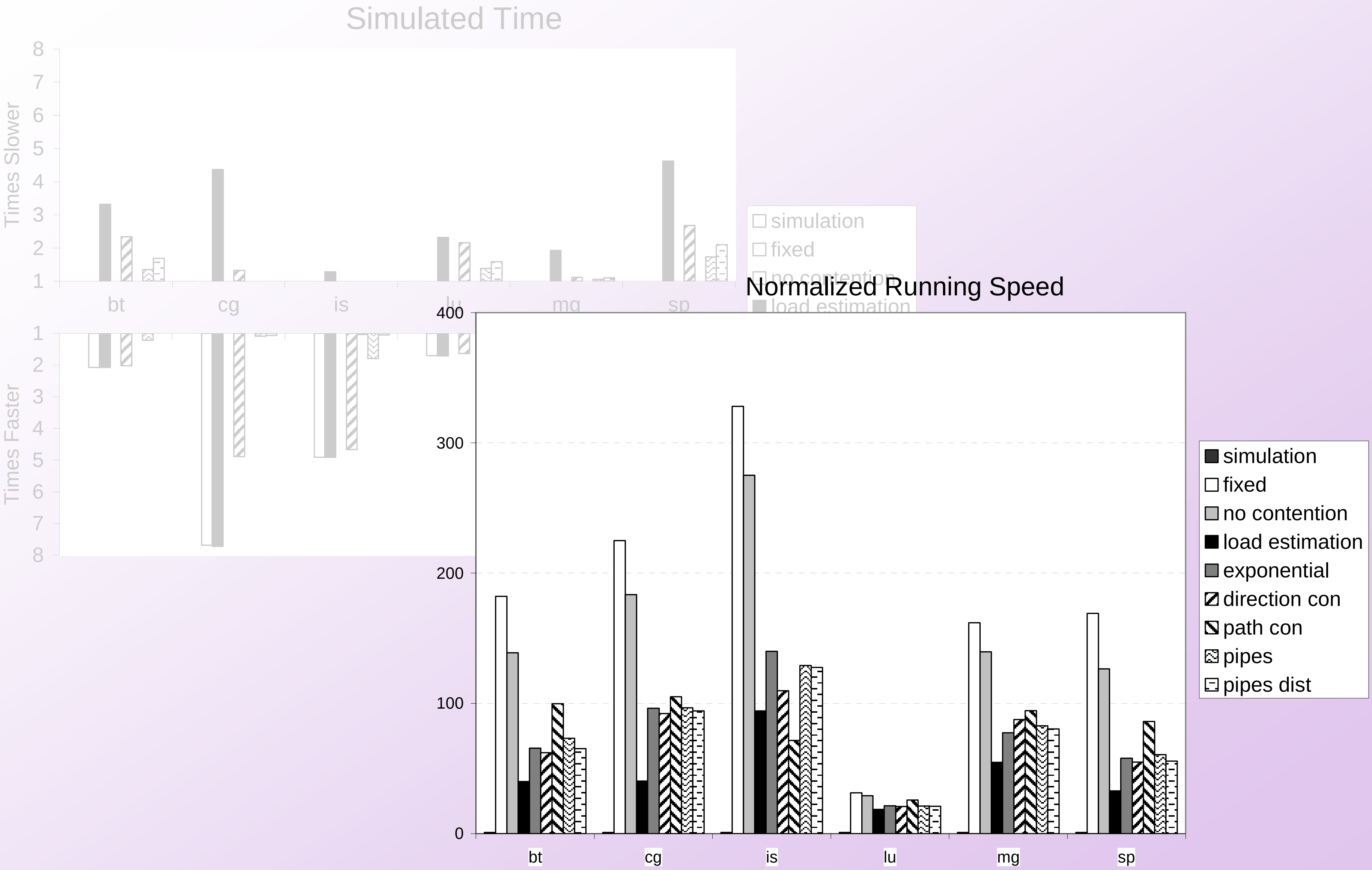
- ⇨ Structured communication patterns
- ⇨ Long messages
- ⇨ States of high congestion



NAS – 64 cores


Simulated Time
Times Slower
8
7
6
5
4
3
2
1
bt
cg
is
lu
mg
sp
1
2
3
4
5
6
7
8
Times Faster
■ simulation
□ fixed
▨ no contention
■ load estimation
■ exponential
▨ direction con
▨ path con
▨ pipes
▨ pipes dist
nning Speed
400
200
100
0
bt
cg
is
lu
mg
sp
□ simulation
□ fixed
□ no contention
■ load estimation
□ exponential
▨ direction con
▨ path con
▨ pipes
▨ pipes dist

NAS – 64 cores


Simulated Time
Times Slower
8
7
6
5
4
3
2
1
bt
cg
is
lu
mg
sp
1
2
3
4
5
6
7
8
Times Faster
□ simulation
□ fixed
□ no contention
■ load estimation
Normalized Running Speed
400
300
200
100
0
bt
cg
is
lu
mg
sp
■ simulation
□ fixed
■ no contention
■ load estimation
■ exponential
▨ direction con
▧ path con
▩ pipes
⊟ pipes dist

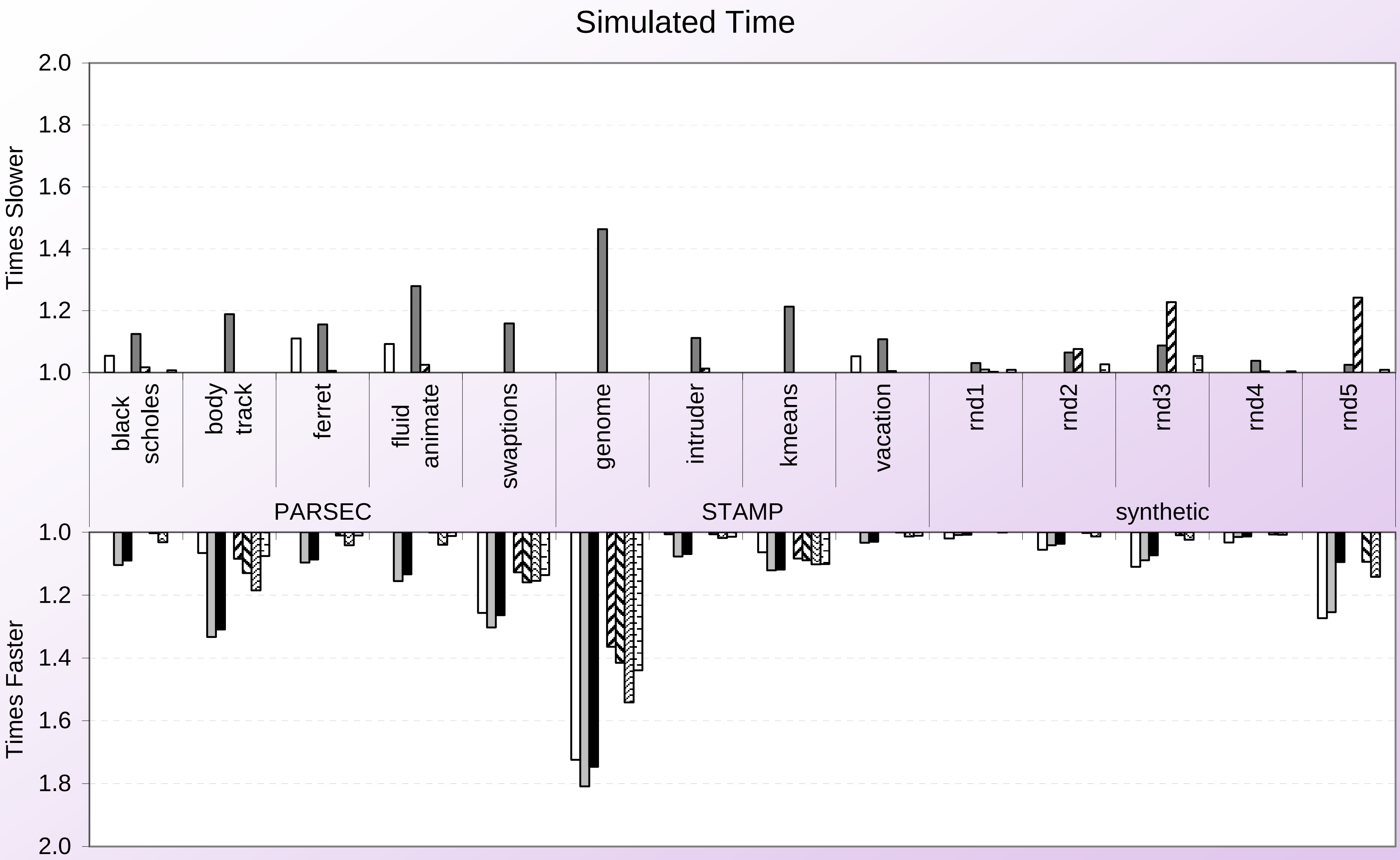
Conclusions

- Novel reservation-based timing models for the NoC
  - Provide reasonable accuracy at a fraction of the speed of a dedicated NoC simulator
- Topology-aware models
  - Considering every link in the topology as a resource provides good accuracy but slows large-scale simulation
  - Modelling a whole direction as a single resource is too restrictive
  - An intermediate approach could be a good solution
- Topology agnostic models
  - Seem to be reasonable models
    - Can be used to discriminate communication-intensive implementations

Future Work

- Implement these models in COTSON
  - Re-evaluate them in this context
- Develop new models for different network configurations based on the reservation data structure
  - Topologies: rings, tori, butterfly, flattened butterfly
  - Packet movement: wormhole, adaptive routing

Other traces results


Simulated Time
Times Slower
Times Faster
black scholes
body track
ferret
fluid animate
swaptions
genome
intruder
kmeans
vacation
rnd1
rnd2
rnd3
rnd4
rnd5
PARSEC
STAMP
synthetic