# Modeling Obstacles in INET/Mobility Framework

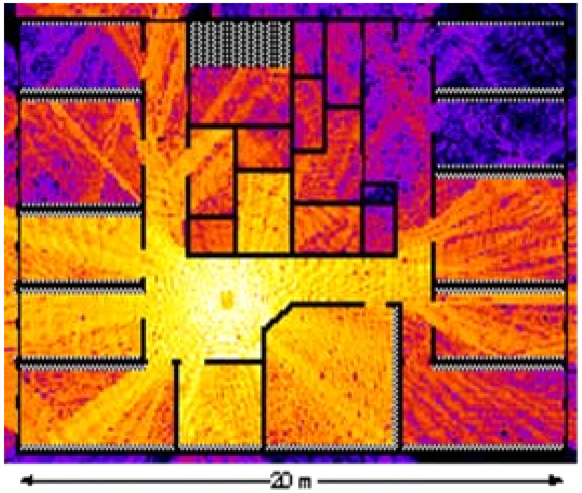
## Motivation, Integration, and Performance

**Hermann S. Lichte**, Jannis Weide
hermann.lichte@upb.de, jannis@mail.upb.de

Computer Networks Group
Universität Paderborn

# How to model obstacles in a wireless network simulation?

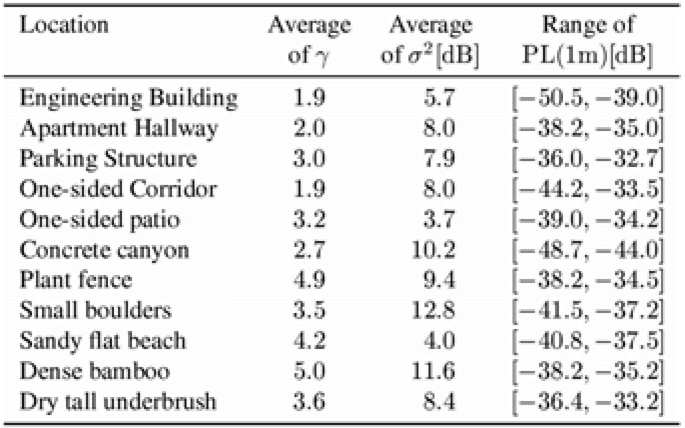
- **Raytracing:** Exact but inefficient, requires detailed model of environment, infeasible for simulation large networks
- **Log-normal shadowing:** Model based on long-term measurements and many obstacles, quick but unspecific



*Raytracing*

**Can we combine both models?**

$$PL(d)[dB] = PL(d_0)[dB] + 10\gamma \log_{10}\left(\frac{d}{d_0}\right) + X_{\sigma^2}$$



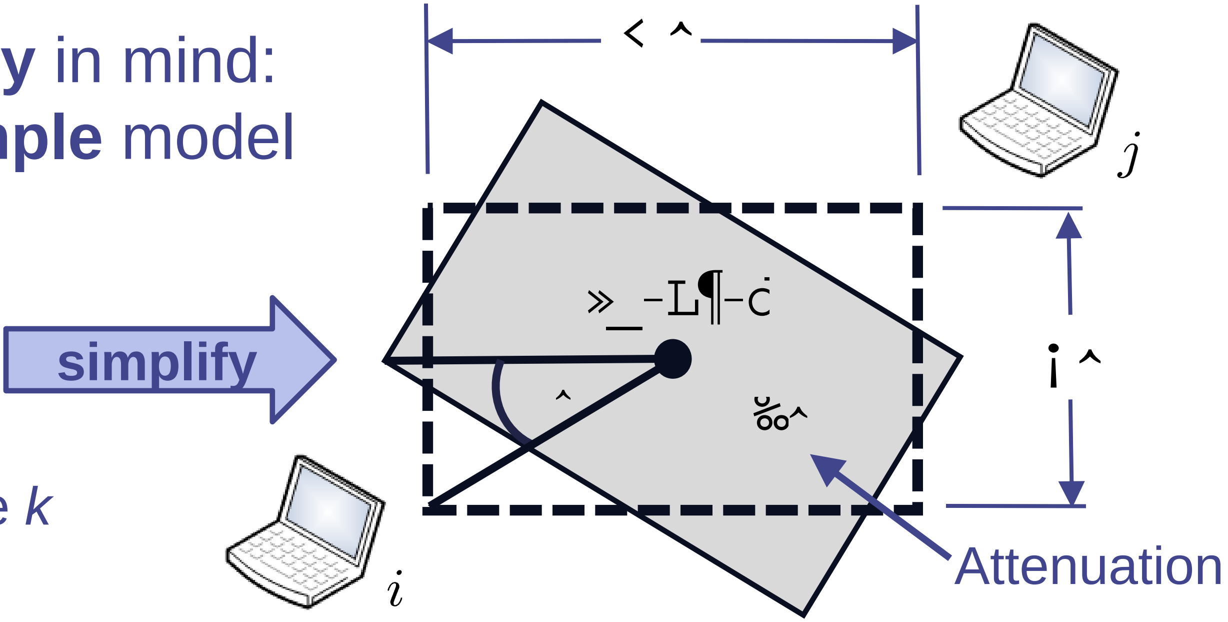
| Location | Average of $\gamma$ | Average of $\sigma^2$[dB] | Range of PL(1m)[dB] |
|---|---|---|---|
| Engineering Building | 1.9 | 5.7 | [−50.5, −39.0] |
| Apartment Hallway | 2.0 | 8.0 | [−38.2, −35.0] |
| Parking Structure | 3.0 | 7.9 | [−36.0, −32.7] |
| One-sided Corridor | 1.9 | 8.0 | [−44.2, −33.5] |
| One-sided patio | 3.2 | 3.7 | [−39.0, −34.2] |
| Concrete canyon | 2.7 | 10.2 | [−48.7, −44.0] |
| Plant fence | 4.9 | 9.4 | [−38.2, −34.5] |
| Small boulders | 3.5 | 12.8 | [−41.5, −37.2] |
| Sandy flat beach | 4.2 | 4.0 | [−40.8, −37.5] |
| Dense bamboo | 5.0 | 11.6 | [−38.2, −35.2] |
| Dry tall underbrush | 3.6 | 8.4 | [−36.4, −33.2] |

*Log-normal shadowing*

# We need a suitable obstacle model

- Keep **efficiency** in mind:
  We need a **simple** model

Some obstacle *k*



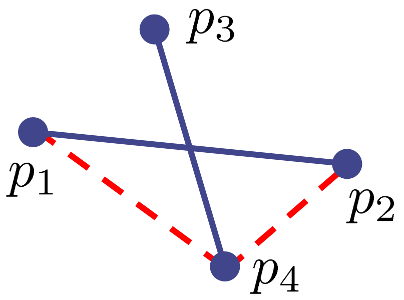
- SINR-based threshold model: $\mathrm{SINR}(i,j) := \dfrac{P_i\, a(i,j)}{N_0 + \sum_{k \neq i} P_k\, a(k,j)} \geq \nu$

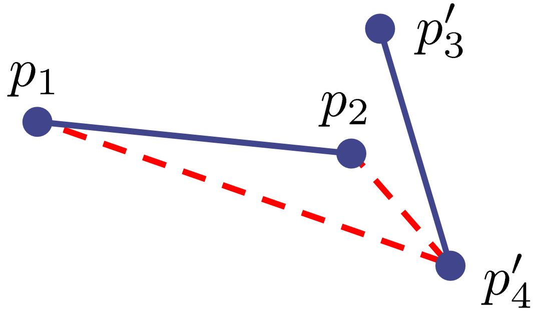
- Channel states: ‰»WL«d I ‰¡ n »WL«d‰W\ »WL«d‰S J »WL«d‰A @ »WL«d

  where ‰A @ »WL«d I 8 U ^ [A ‰ k^W« and k^W« I ƒ ÐpDq|0Vhjphqwv0Lqwhuvhfw+c^W«, H b<y"/Dr"

# ANY-SEGMENTS-INTERSECT for two line segments

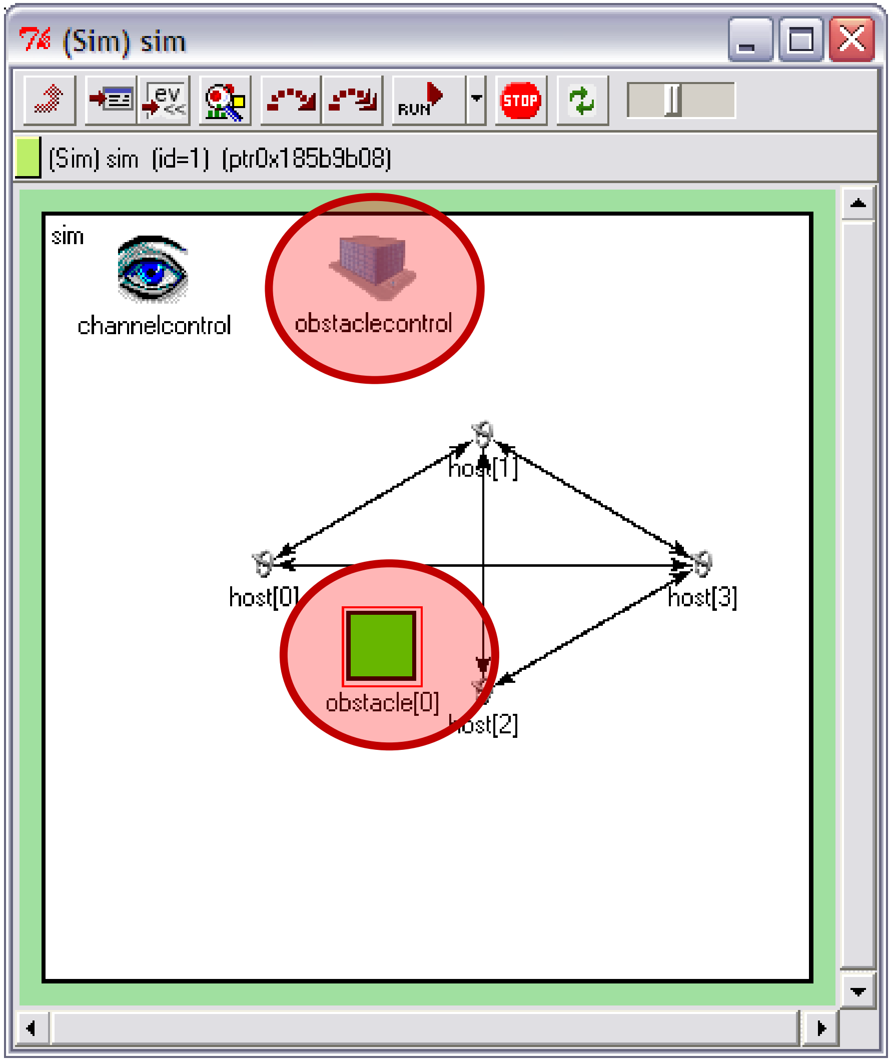
- $p_1$ and $p_2$ are on opposite sides of $\overline{p_3 p_4}$
- $p_3$ and $p_4$ are on opposite sides of $\overline{p_1 p_2}$



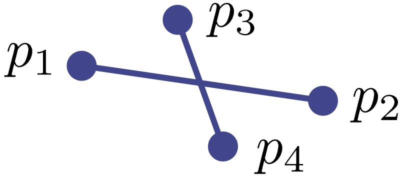


- Sign of cross-product determines orientation
  - i.e. $\operatorname{sign}(\overline{p_4 p_1} \times \overline{p_3 p_4}) \neq \operatorname{sign}(\overline{p_4 p_2} \times \overline{p_3 p_4})$
  - and $\operatorname{sign}(\overline{p_3 p_1} \times \overline{p_1 p_2}) \neq \operatorname{sign}(\overline{p_3 p_2} \times \overline{p_1 p_2})$
- Also check whether endpoints coincide with line segments

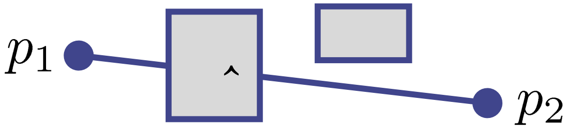
# New features – ObstacleControl and its methods



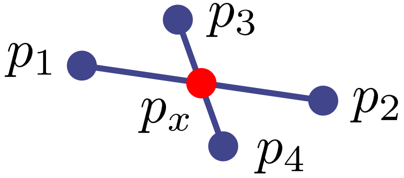
- **testIntersect()** – true/false



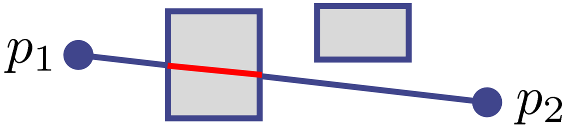
- **calcObstacleDecrease()** and **getIntersectingObstacles()**



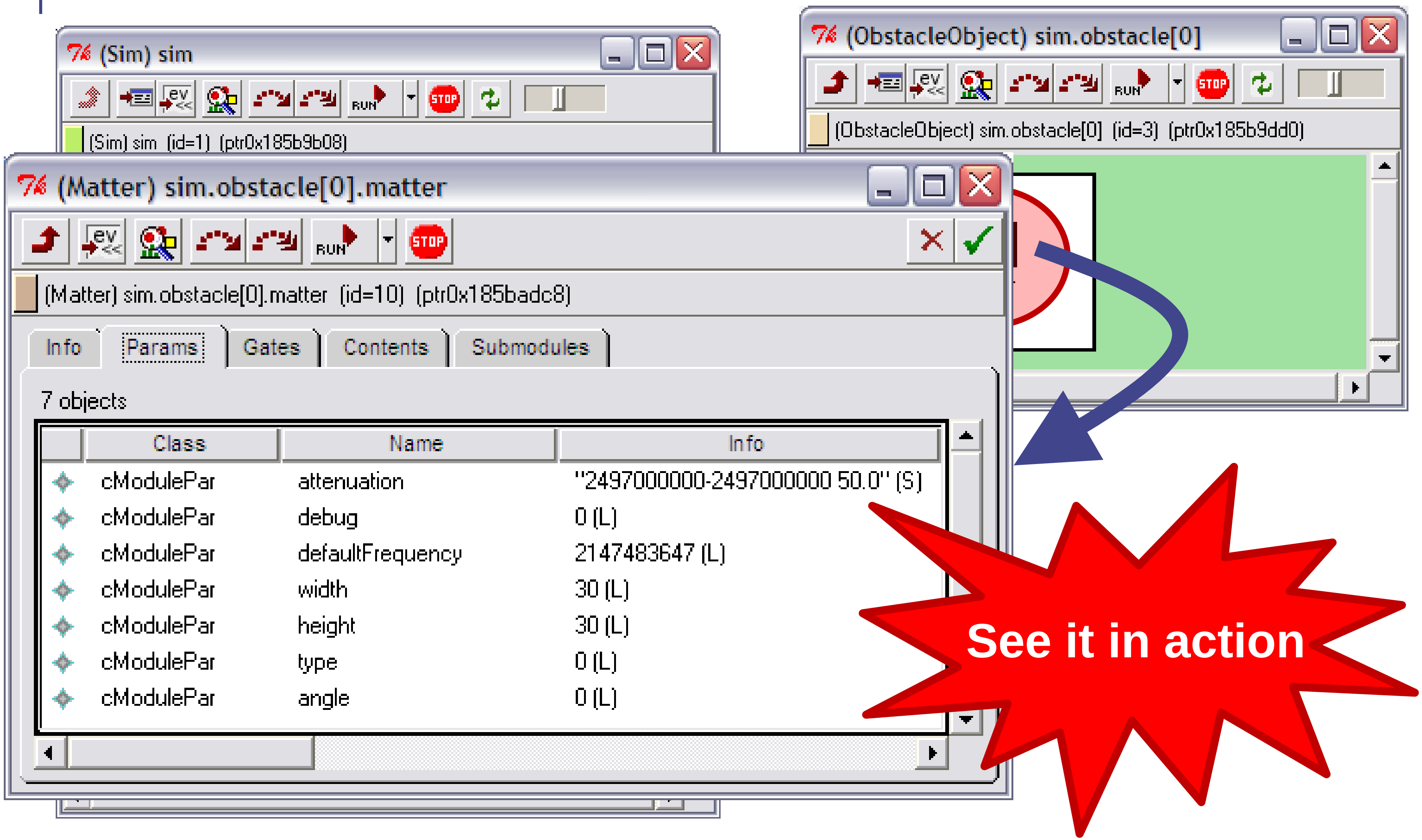
- **getIntersection()** – point $p_x$



- **getIntersectionLength()**



# New features – Matter and its attributes



(Sim) sim
RUN
STOP
(ObstacleObject) sim.obstacle[0]
(Matter) sim.obstacle[0].matter
Info
Params
Gates
Contents
Submodules

(Sim) sim (id=1) (ptr0x185b9b08)

(ObstacleObject) sim.obstacle[0] (id=3) (ptr0x185b9dd0)

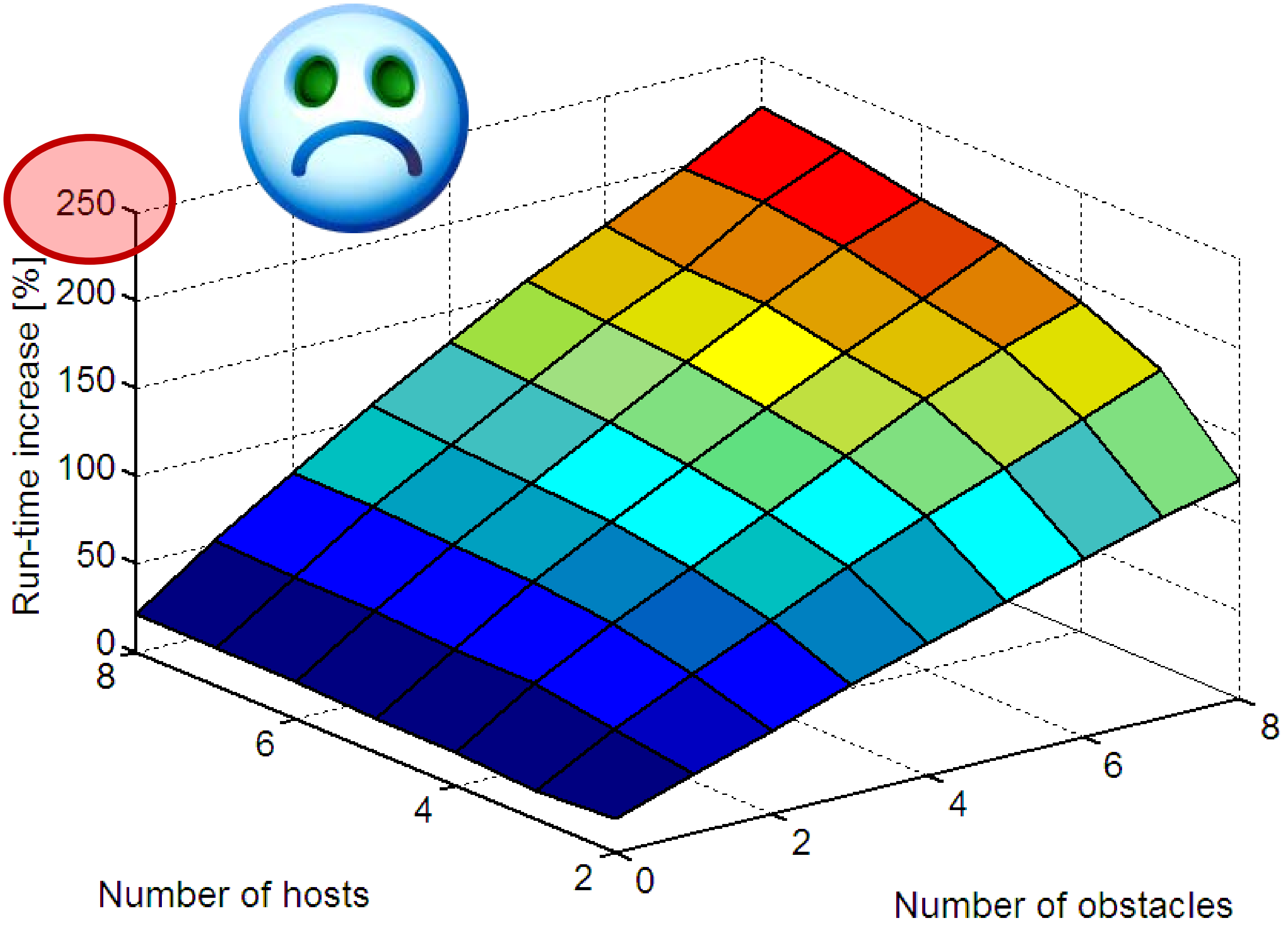
(Matter) sim.obstacle[0].matter (id=10) (ptr0x185badc8)

7 objects

| Class | Name | Info |
|---|---|---|
| cModulePar | attenuation | "2497000000-2497000000 50.0" (S) |
| cModulePar | debug | 0 (L) |
| cModulePar | defaultFrequency | 2147483647 (L) |
| cModulePar | width | 30 (L) |
| cModulePar | height | 30 (L) |
| cModulePar | type | 0 (L) |
| cModulePar | angle | 0 (L) |

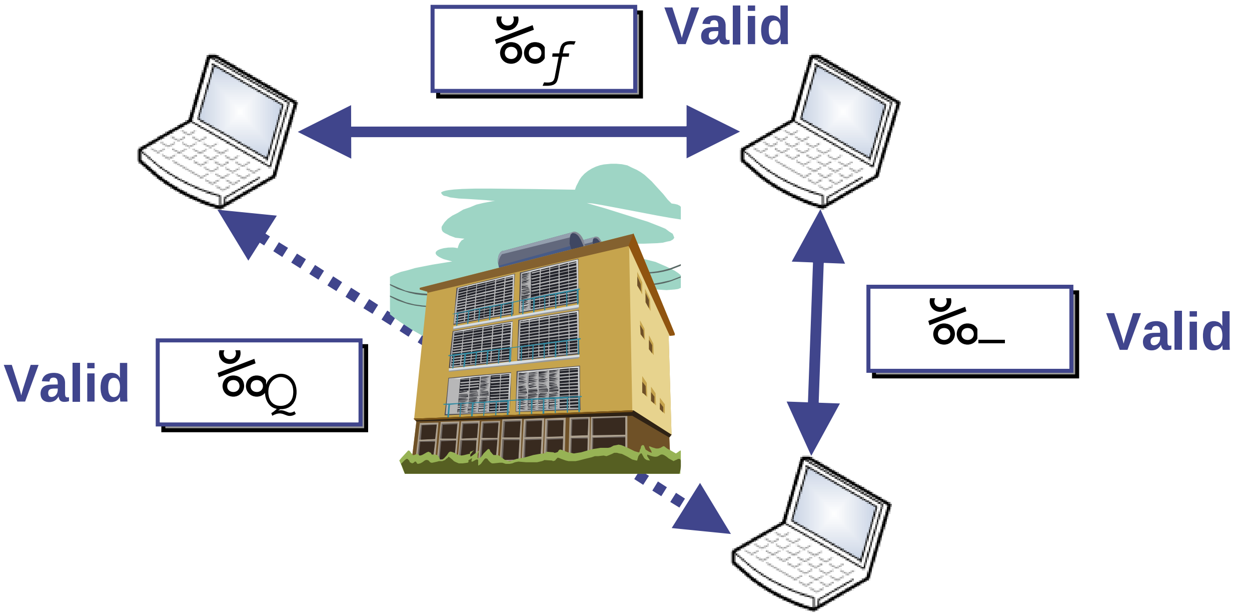
See it in action

# Run-time increase of a trivial extension



# Improving run-time by connection caches

- Checking for intersections is time-consuming
- **Idea:** Cache attenuation factors per connection



‰f

Valid

‰_

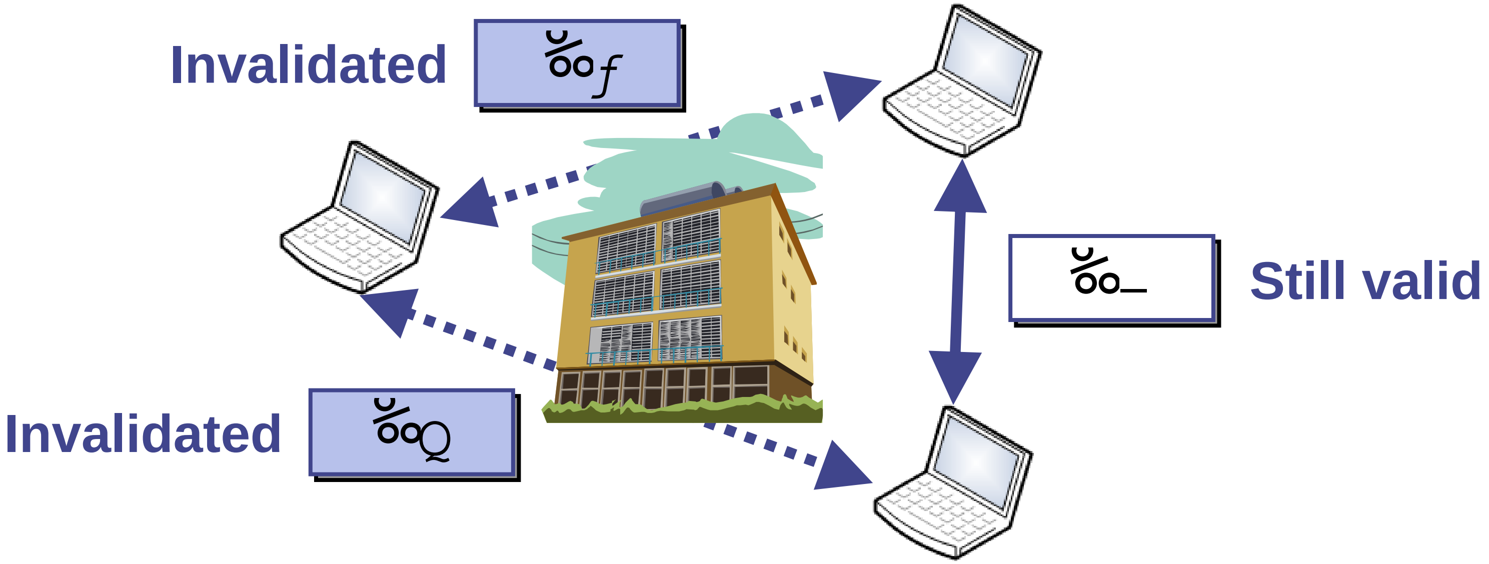
Valid

Valid

‰Q

# Improving run-time by connection caches

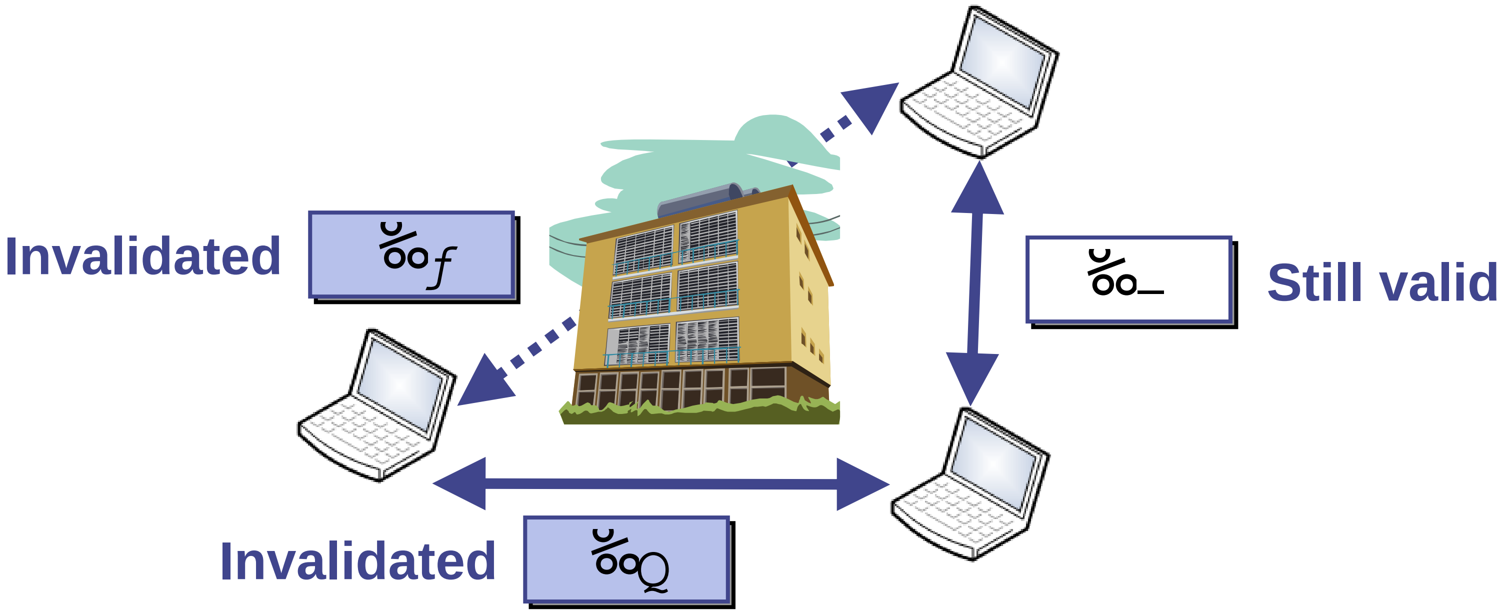
- Checking for intersections is time-consuming
- **Idea:** Cache attenuation factors per connection



- Invalidate all connections that are affected by the move

# Improving run-time by connection caches

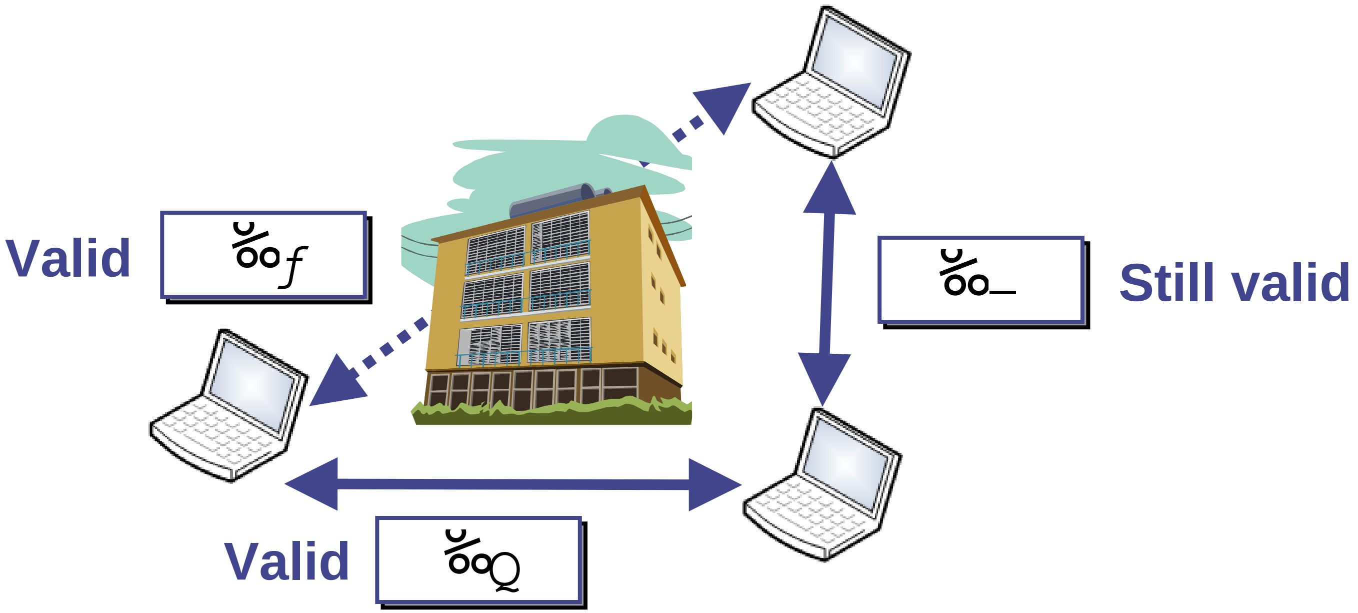
- Checking for intersections is time-consuming
- **Idea:** Cache attenuation factors per connection



- Invalidate all connections that are affected by the move

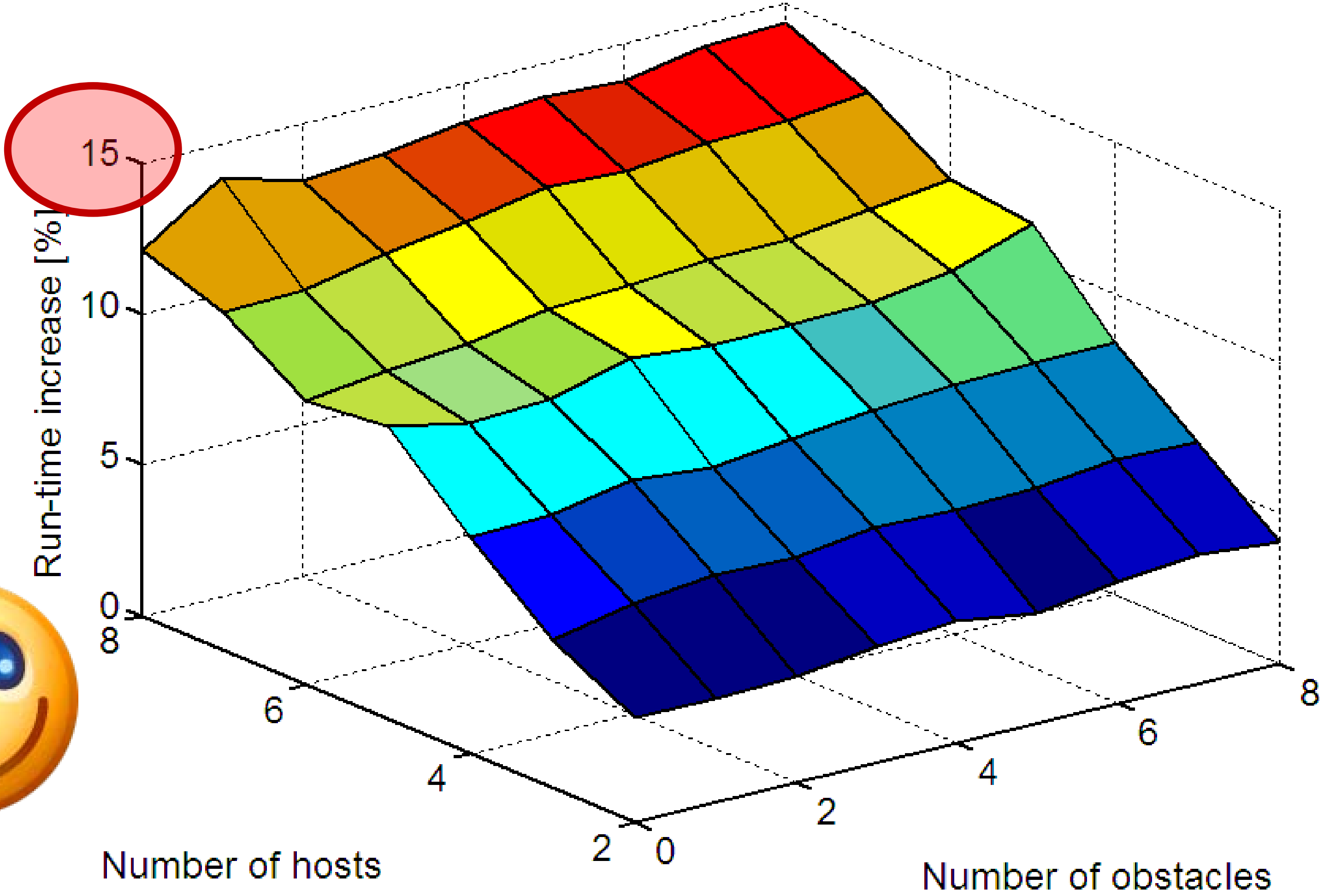
# Improving run-time by connection caches

- Checking for intersections is time-consuming
- **Idea:** Cache attenuation factors per connection



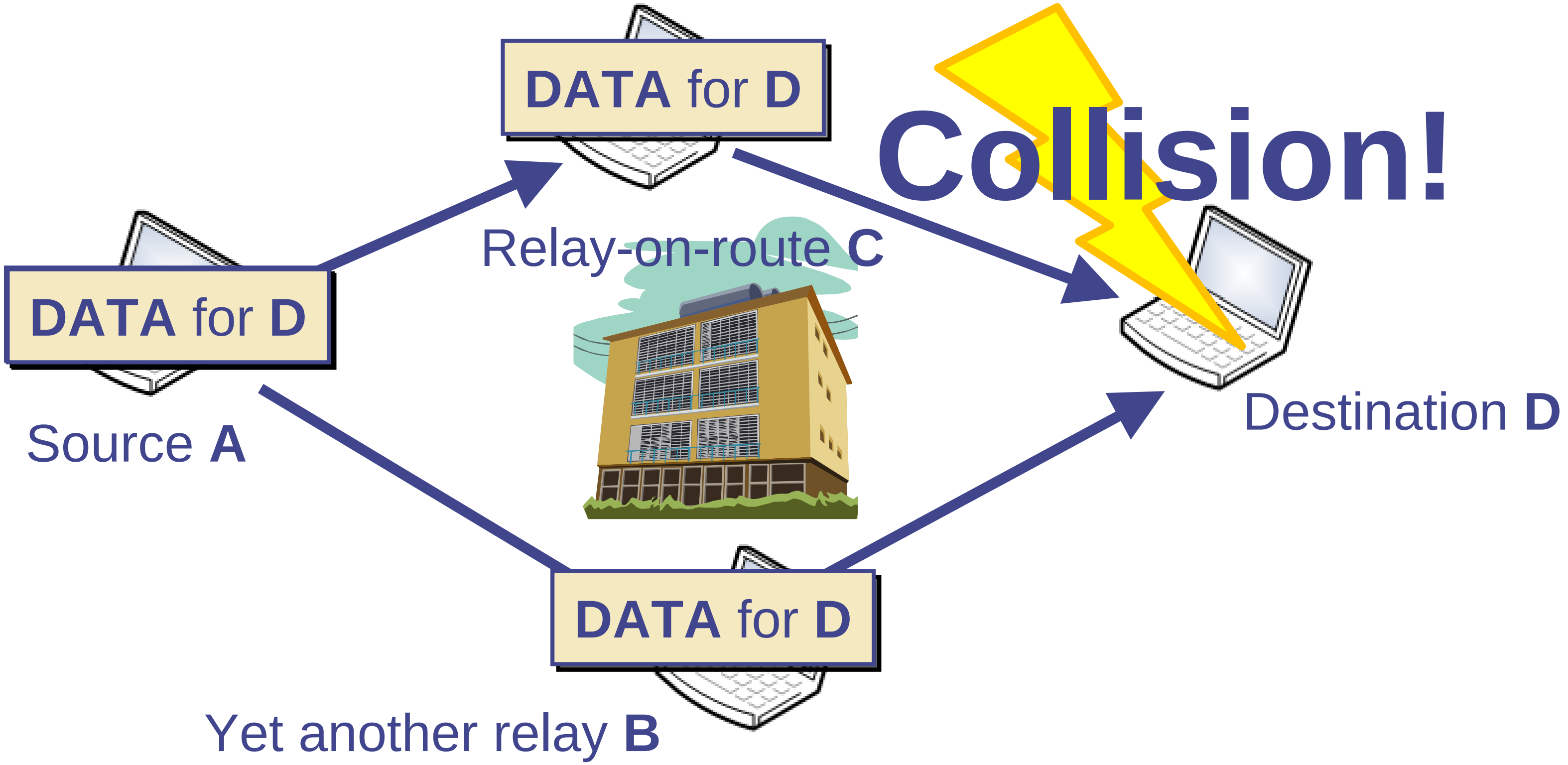
- Invalidate all connections that are affected by the move

# Run-time increase when connection caches used



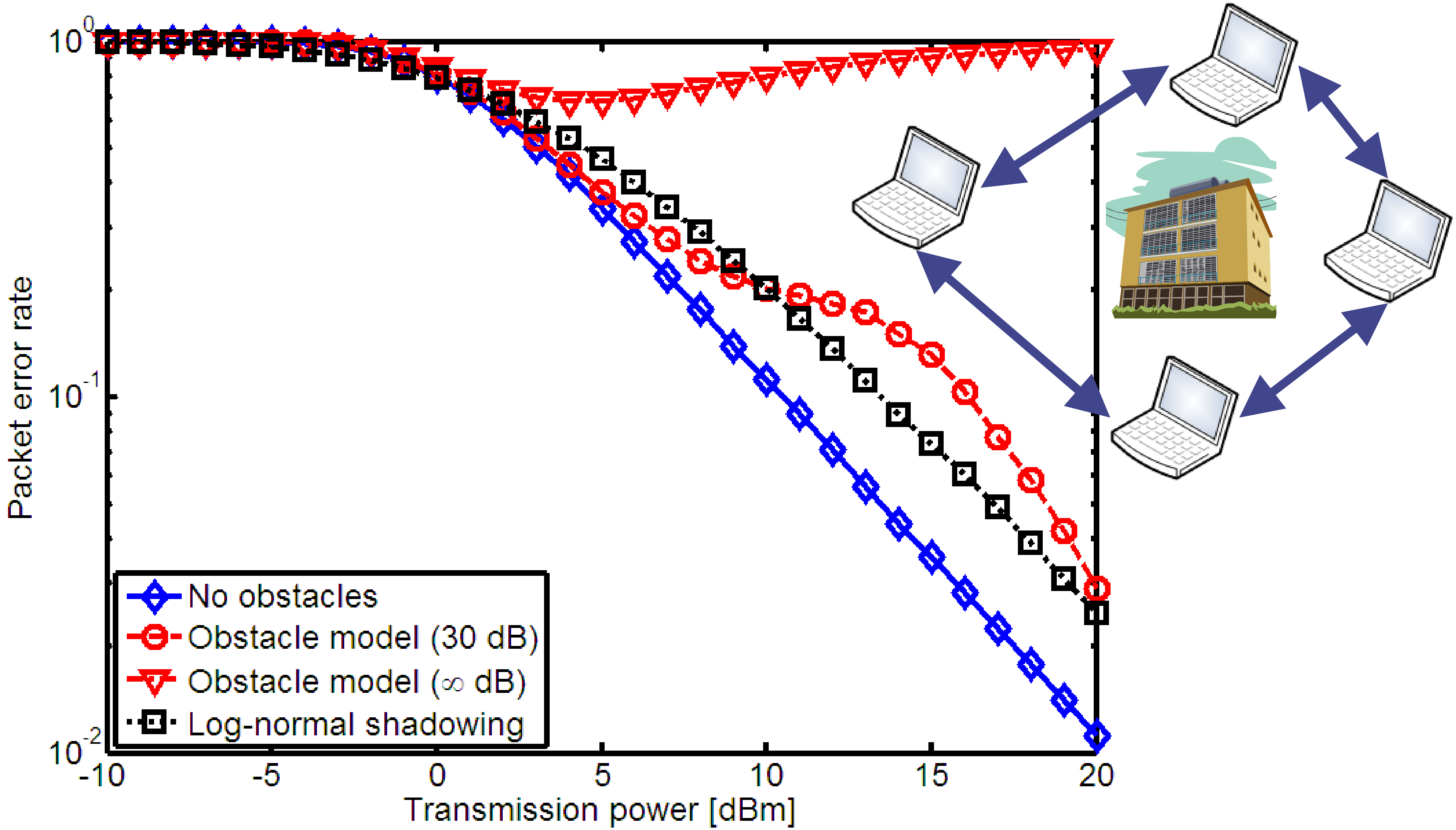
# Example – Protocol evaluation with carrier sensing

- Wireless multi-hop transmission from **A** via **C** to **D**
- **B** retransmits when it cannot overhear **C**

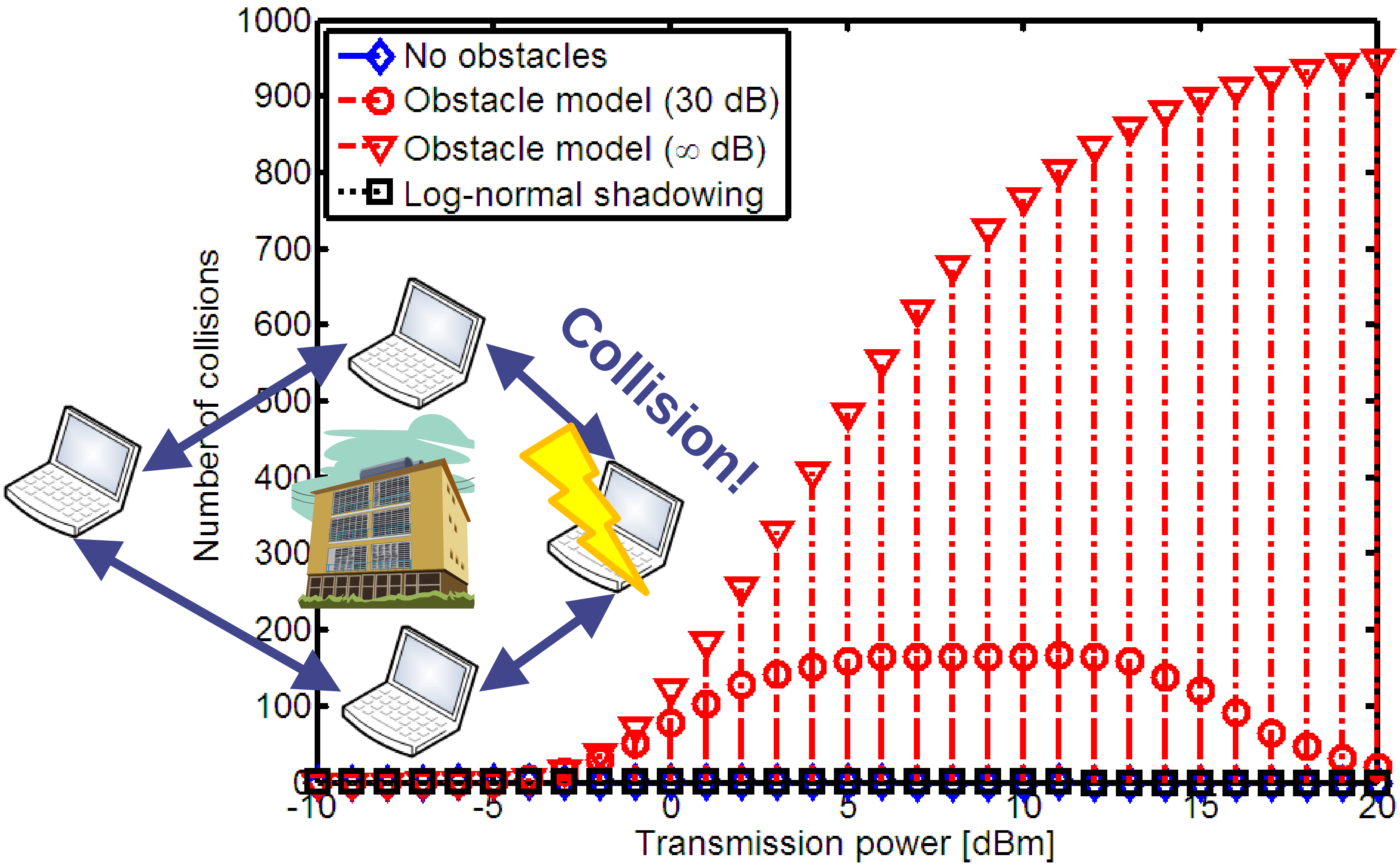


Obstacles may cause unexpected behavior in your protocol

# Obstacle model helps to reveal bad error rates

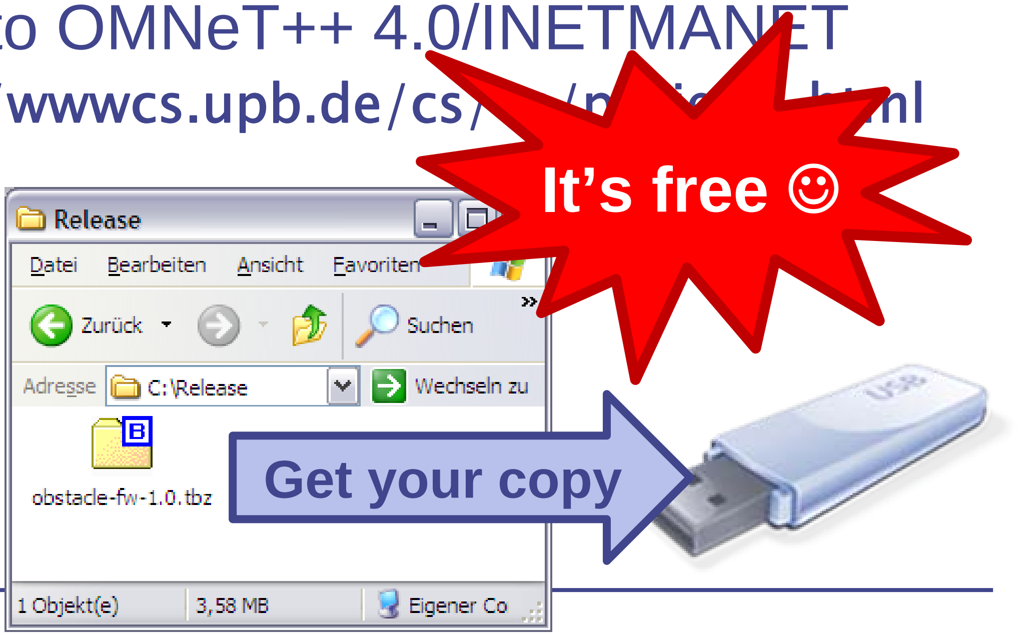


# Error rates caused by collisions due to shielding



# Conclusions

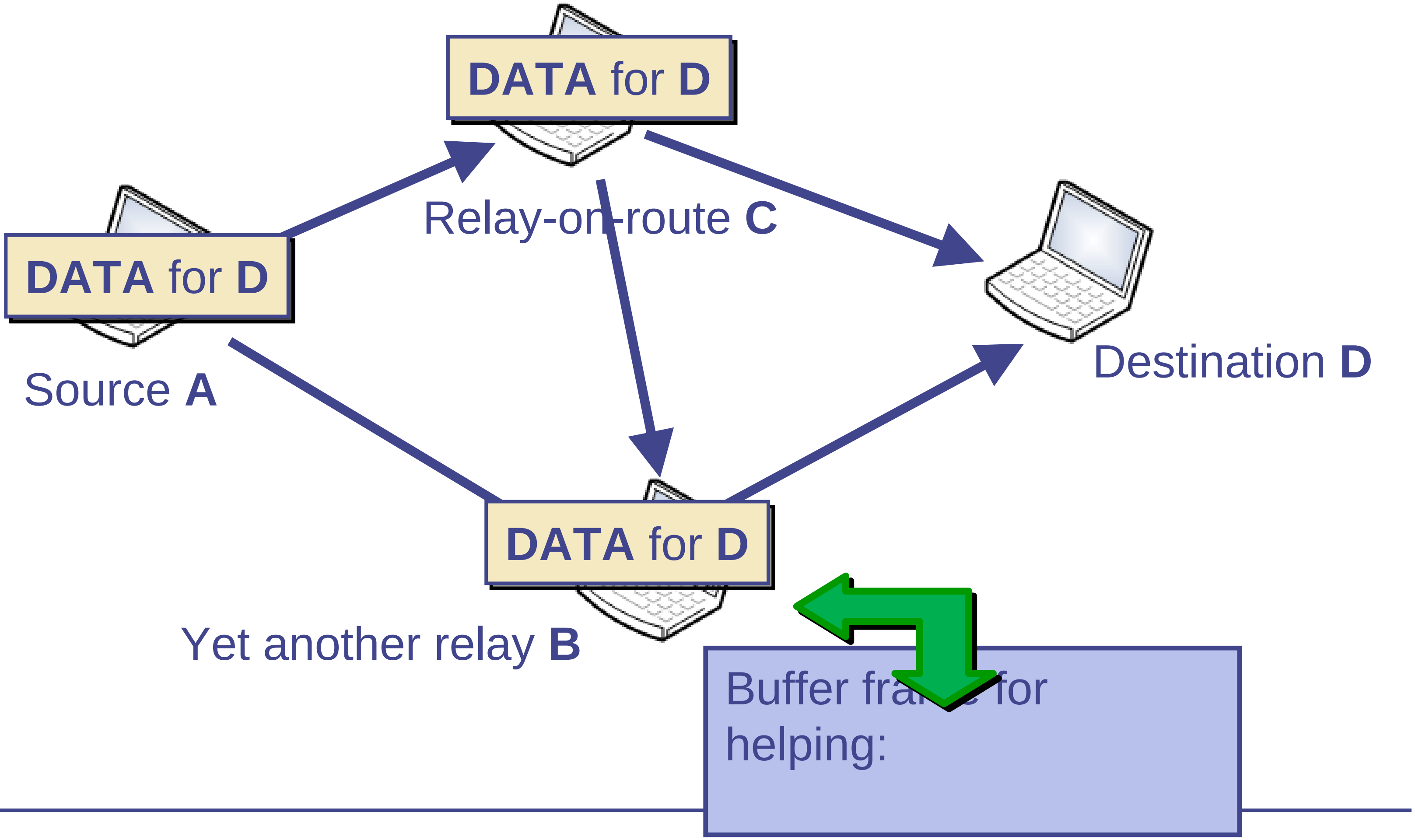
- Inadequate models may not capture “true” performance
- The obstacle model proposed today:
  - Captures shielding effects
  - Efficient and easy to use
  - Add-on for stochastic models
- **<u>Caching is essential for simulation’s performance</u>**
- We’re currently porting it to OMNeT++ 4.0/INETMANET
  - Will be released at **http://wwwcs.upb.de/cs/[illegible]/[illegible]ml**
- **Get your copy now to**
  - Play with it
  - Reproduce results
  - Port to other frameworks
  - Improve it



- Backup slides

# You made up a cool protocol that you want to test

- Wireless multi-hop transmission from **A** via **C** to **D**



**DATA** for **D**

Relay-on-route **C**

**DATA** for **D**

Source **A**

Destination **D**

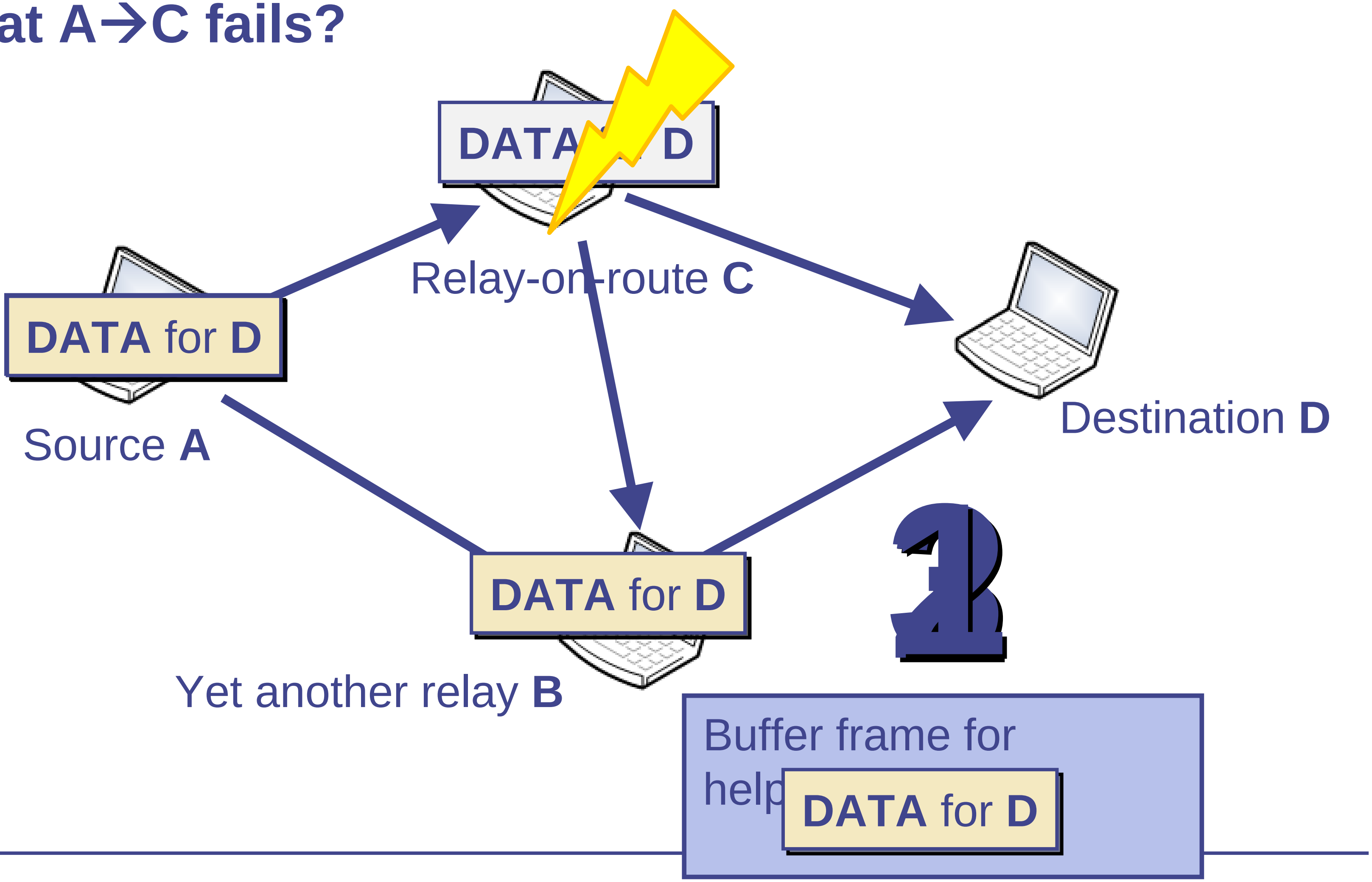
**DATA** for **D**

Yet another relay **B**

Buffer frame for helping:

# You made up a cool protocol that you want to test

- Wireless multi-hop transmission from **A** via **C** to **D**
- **What A→C fails?**



DATA for D
DATA for D
Relay-on-route **C**
DATA for D
Destination **D**
Source **A**
DATA for D
Yet another relay **B**
2
1
Buffer frame for help
DATA for D