Lifetime-aware Design Methodology for Dynamic Partially Reconfigurable Systems

Presented by:
Siva Satyendra Sahoo

Siva Satyendra Sahoo, Dr. Bharadwaj Veeravalli
Department of Electrical and Computer Engineering, National University of Singapore
satyendra@u.nus.edu, elebv@nus.edu.sg

Dr. Tuan D.A. Nguyen , Dr. Akash Kumar
Center for Advancing Electronics Design, Technische Universitat, Dresden
tuan_duy_anh.nguyen1,akash.kumar@tu-dresden.de

Outline

- Motivation
- Dynamic Partial Reconfiguration (DPR):
  - Background
  - Features
  - Aging mitigation
- System model
- System Design methodology
- Experiment and Results
- Conclusion

Outline

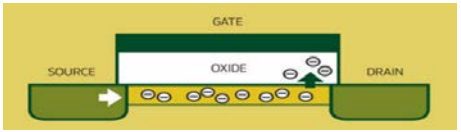
- **Motivation**
- Dynamic Partial Reconfiguration:
  - Background
  - Features
  - Aging mitigation
- System model
- System Design methodology
- Experiment and Results
- Conclusion

Increasing fault-rates

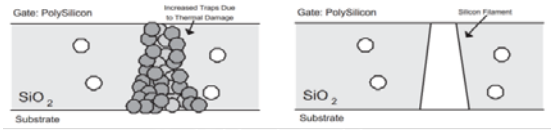
Physical faults

- Intermittent faults
- Permanent faults

Mechanisms:

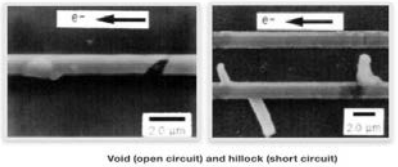


Hot carrier injection

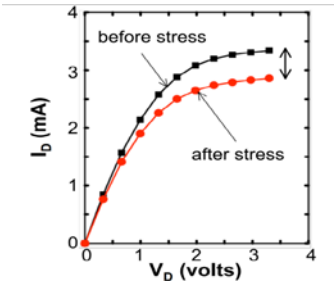


Hard Dielectric Breakdown

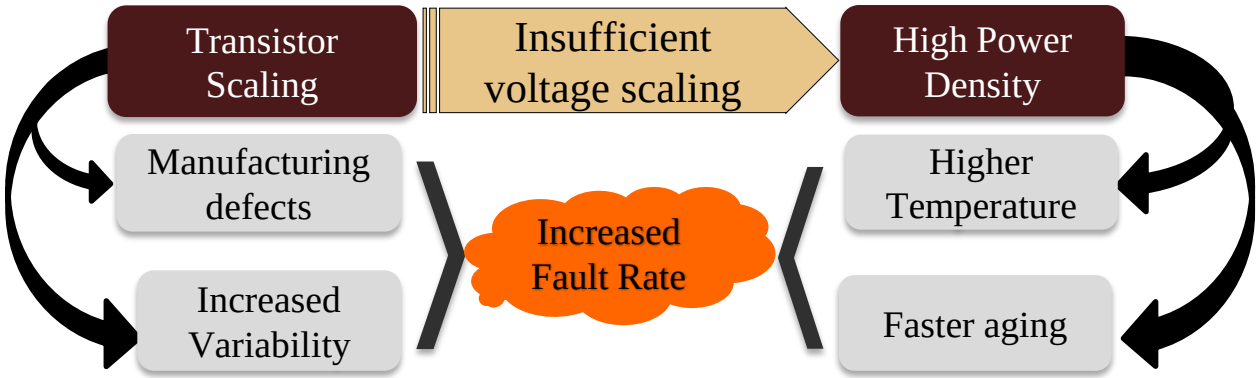
Gate-oxide Breakdown



Electromigration



Negative Bias Temperature Instability



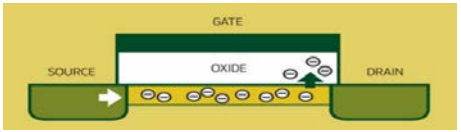
J. Keane and C. H. Kim, "An odometer for CPUs," in IEEE Spectrum, vol. 48, no. 5, pp. 28-33, May 2011.
J. Srinivasan, S. V. Adve, P. Bose and J. A. Rivers, "The impact of technology scaling on lifetime reliability," Dependable Systems and Networks, 2004 International Conference on, 2004, pp. 177-186.

Increasing fault-rates

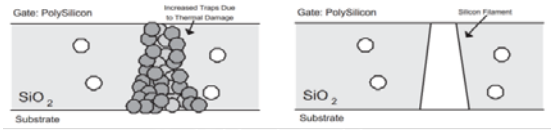
Physical faults

- **Intermittent faults**
- **Permanent faults**

Mechanisms:

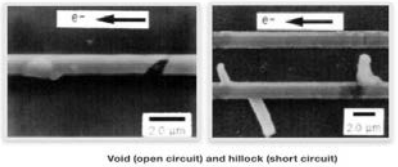


Hot carrier injection



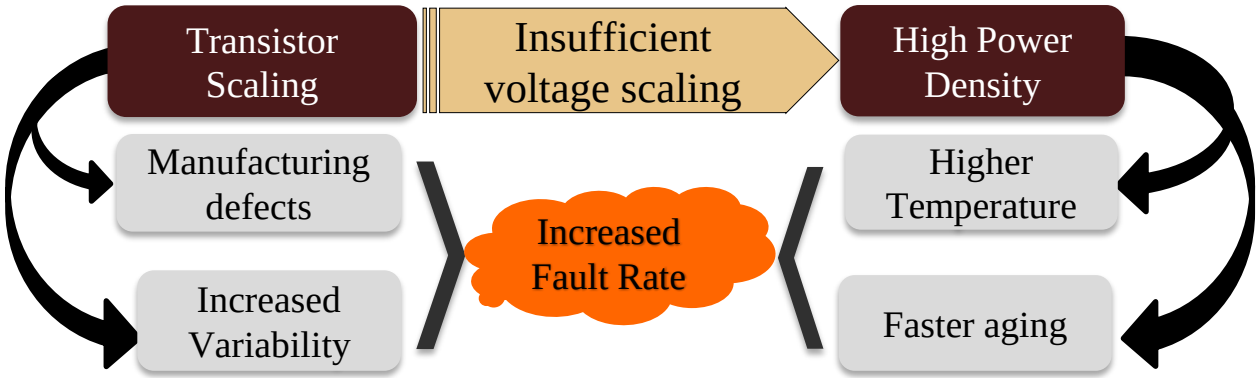
Hard Dielectric Breakdown

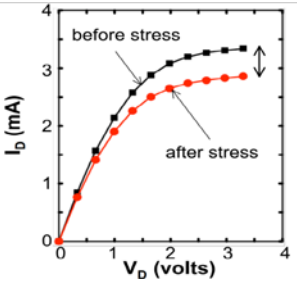
Gate-oxide Breakdown



Electromigration

Reduced System Lifetime



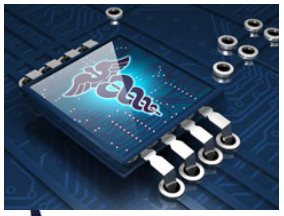


Negative Bias Temperature Instability

J. Keane and C. H. Kim, "An odometer for CPUs," in IEEE Spectrum, vol. 48, no. 5, pp. 28-33, May 2011.
J. Srinivasan, S. V. Adve, P. Bose and J. A. Rivers, "The impact of technology scaling on lifetime reliability," Dependable Systems and Networks, 2004 International Conference on, 2004, pp. 177-186.

System-level effect

- Mission failures

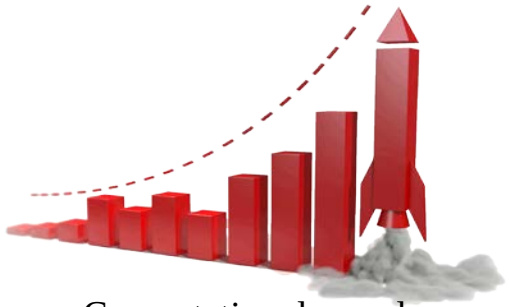


- Reduced safety in critical systems
  - Power plants, transportation, medical etc.

- Reduced product lifetime

Run-time Resource Sharing

- **Meeting increasing computation demands:**
  - Parallelism
  - Custom Computing
    - *Hardware Accelerators*

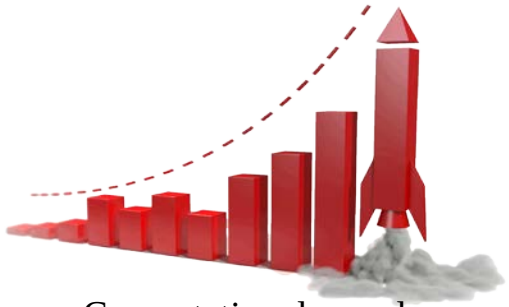


Computation demands

http://www.responsiblelending.org/
http://www.chip-architect.com/

Run-time Resource Sharing

- **Meeting increasing computation demands:**
  - Parallelism
  - Custom Computing
    - *Hardware Accelerators*



Computation demands

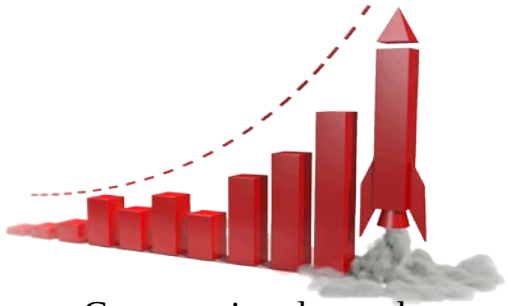
Finite computation resources !!

- ***Time-sharing* of computing resources:**
  - Cost-efficient *parallel* systems

http://www.responsiblelending.org/
http://www.chip-architect.com/

Run-time Resource Sharing

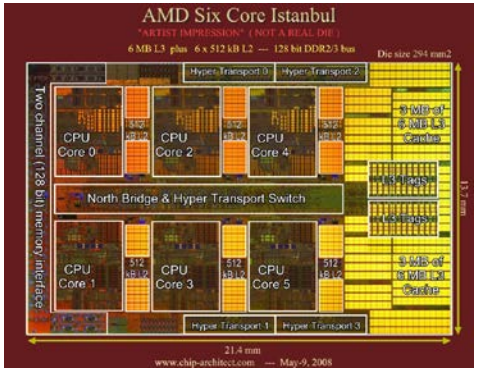
- **Meeting increasing computation demands:**
  - Parallelism
  - Custom Computing
    - *Hardware Accelerators*



Computation demands

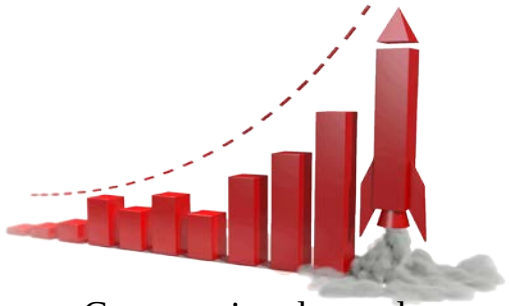
Finite computation resources !!

- ***Time-sharing* of computing resources:**
  - Cost-efficient *parallel* systems
  - Multi-processor and/or Multi-core SoCs:
    - Multiple applications sharing a number of *instruction-set processor pipeline*



Run-time Resource Sharing

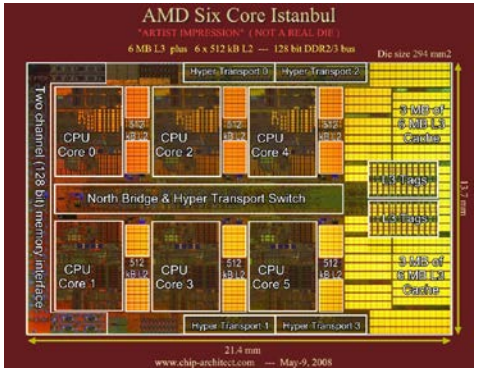
- **Meeting increasing computation demands:**
  - Parallelism
  - Custom Computing
    - *Hardware Accelerators*



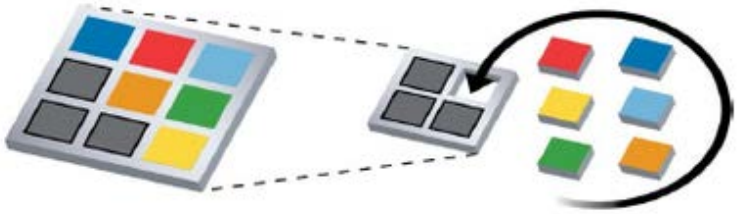
Computation demands

Finite computation resources !!

- ***Time-sharing* of computing resources:**
  - Cost-efficient *parallel* systems
  - Multi-processor and/or Multi-core SoCs:
    - Multiple applications sharing a number of *instruction-set processor pipeline*
  - **FPGAs:**
    - Multiple hardware accelerators sharing *reconfigurable hardware*
    - *Parallel* + *"Custom"*

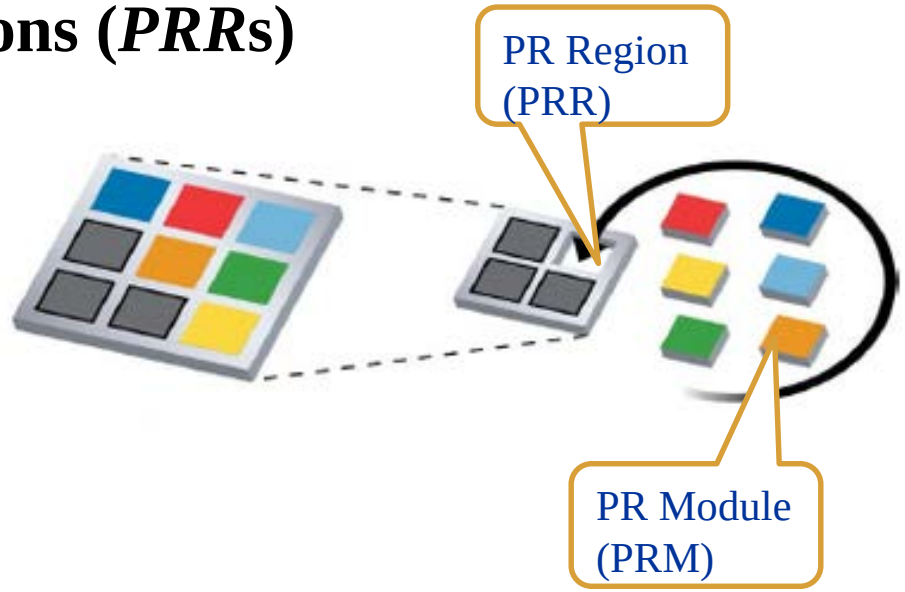


DPR



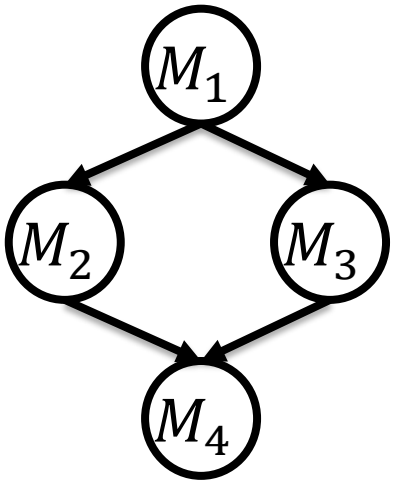
Dynamic Partial Reconfiguration (*DPR*)

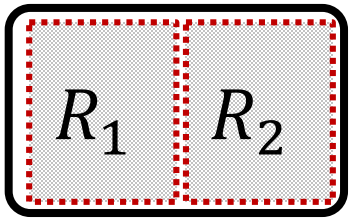
- **Partially Reconfigurable Modules (*PRM*s)**
- **Partially Reconfigurable Regions (*PRR*s)**

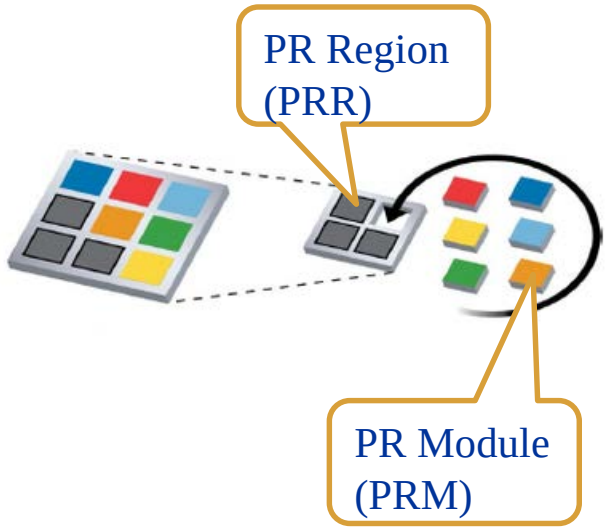


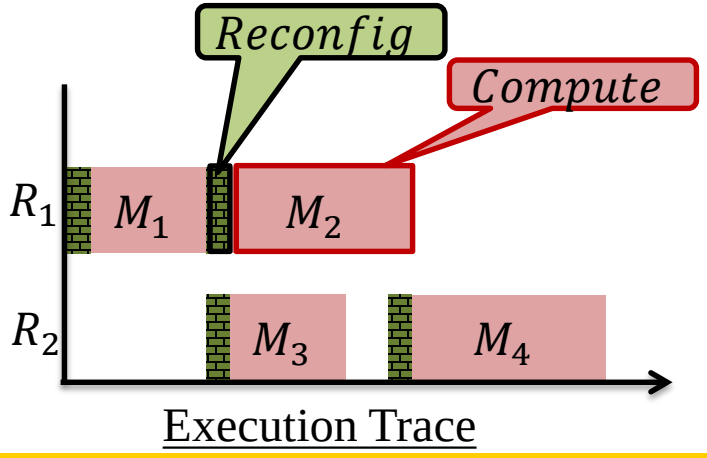
Dynamic Partial Reconfiguration (*DPR*)

- **Partially Reconfigurable Modules (*PRMs*)**
- **Partially Reconfigurable Regions (*PRRs*)**





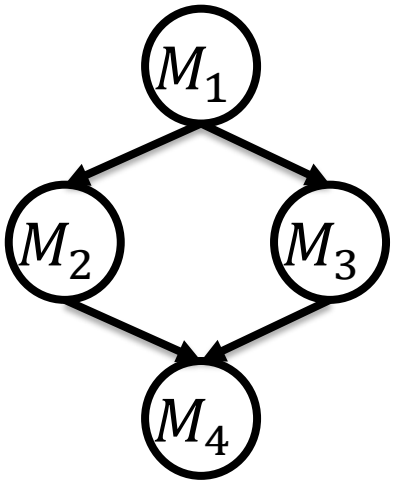


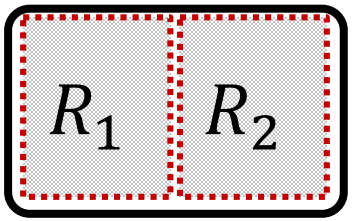


Outline

- Motivation
- **Dynamic Partial Reconfiguration:**
  - Background
  - **Features**
  - Aging mitigation
- System model
- System Design methodology
- Experiment and Results
- Conclusion

PRM-PRR *compatibility*





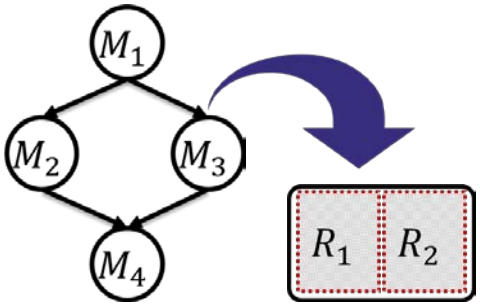
PRRs → / PRMs ↓	R_1	R_2
M_1	✓	✓
M_2	✓	✗
M_3	✗	✓
M_4	✓	✓

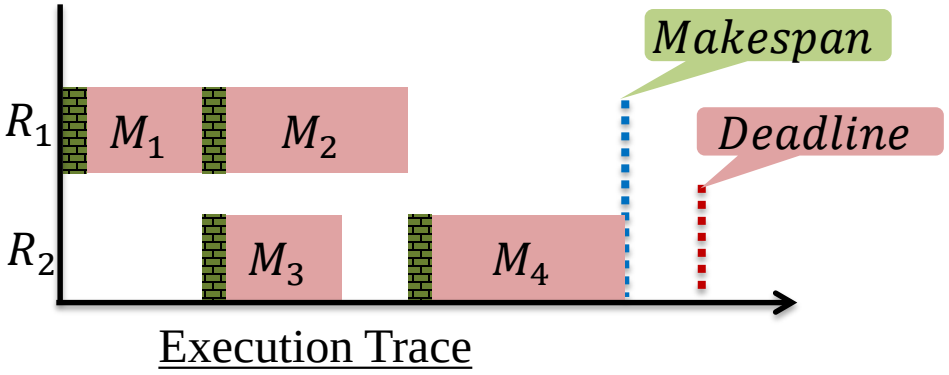
PRM-PRR Compatibility

Affects:
- PRRs' *size*
- #PRRs (*available* parallelism)
- Bitstreams *storage*

Scheduling PRMs on PRRs

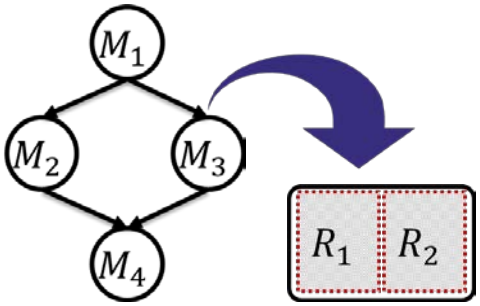
- **Deadlines**
  - *Latency* (*Timing* Reliability)

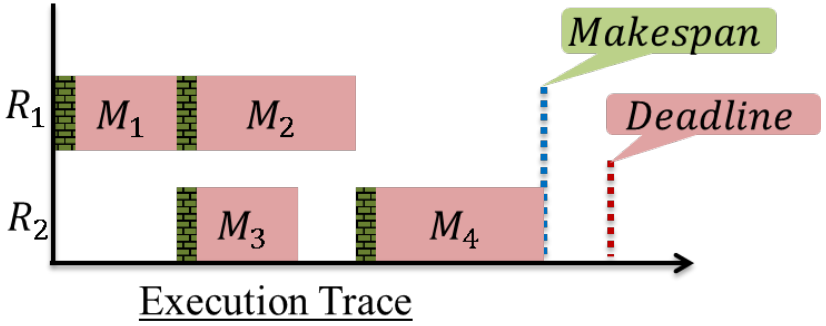


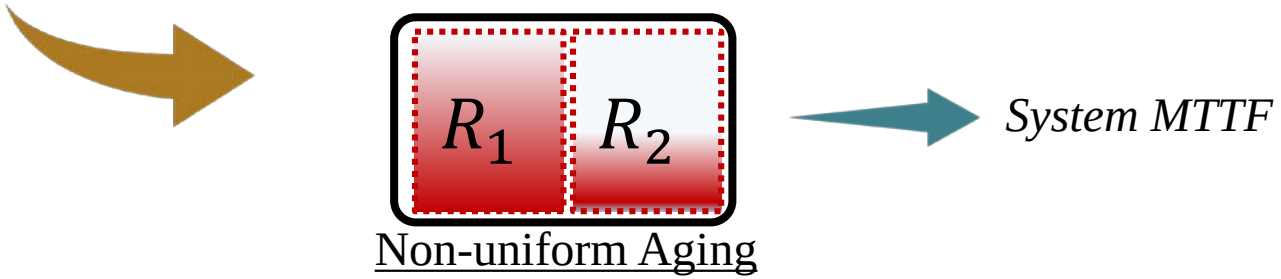


Scheduling PRMs on PRRs

- Deadlines
  - *Latency* (*Timing* Reliability)
- **Aging**
  - *System MTTF* (*Lifetime* Reliability)

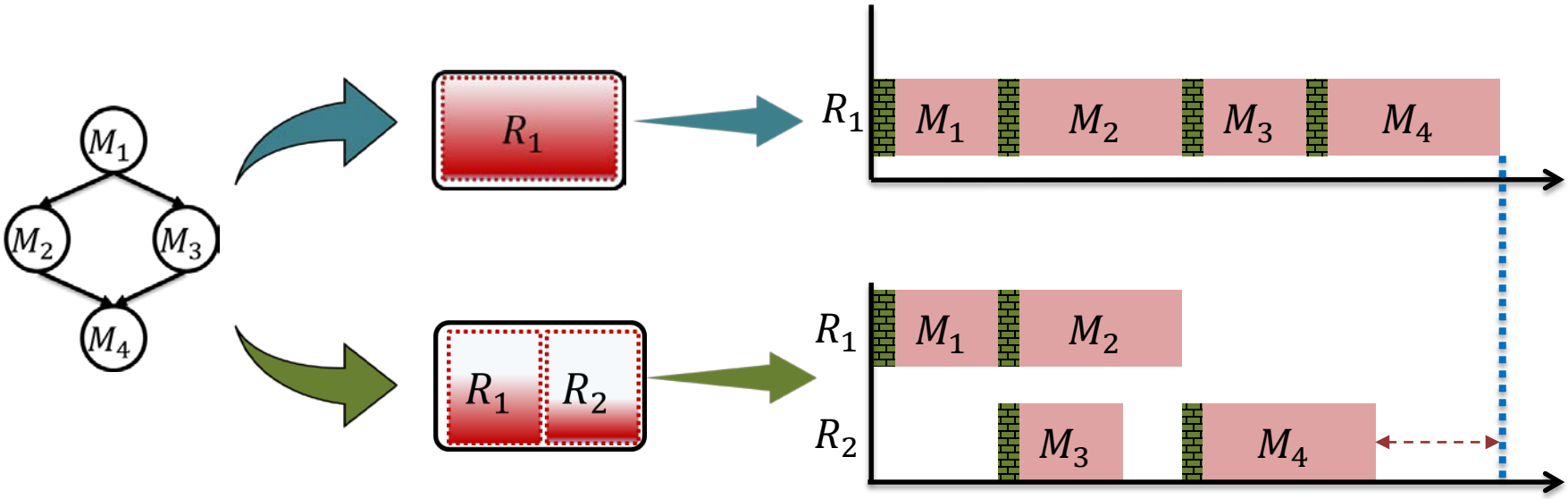






System-level *Spatial Redundancy*

- **Number of available PRRs**
  - Increased available *parallelism*
  - *Net* aging reduced



DPR and System Lifetime

- **PRM-PRR *compatibility***
- **Scheduling : *Deadlines* and *Aging***
- **System-level *Spatial Redundancy***

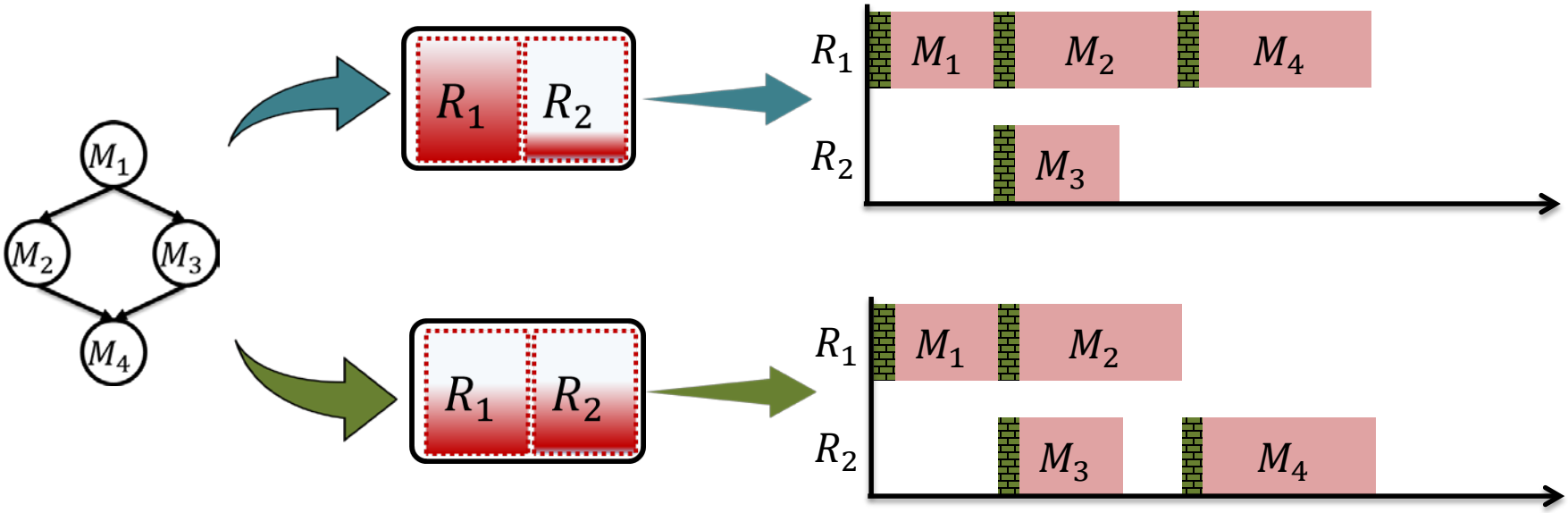
Tools to improve the system MTTF

Outline

- Motivation
- **Dynamic Partial Reconfiguration**:
  - Background
  - Features
  - **Aging mitigation**
- System model
- System Design methodology
- Experiment and Results
- Conclusion

System MTTF-aware *Scheduling*

- **Aim: Reduce aging of each PRR**
- **Constraints:**
  - **Execution latency**

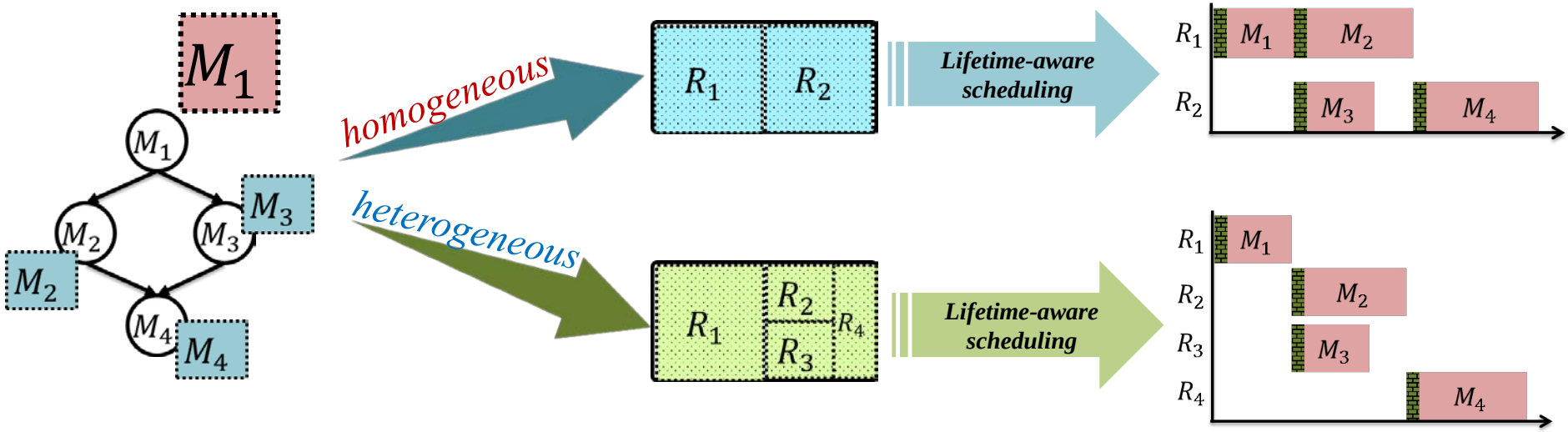


System MTTF-aware system *partitioning*

- ***Homogeneous* v/s *Heterogeneous* PRRs**

  Effects:

  - ***Maximum #PRRs***
  - ***Aging of each PRR***

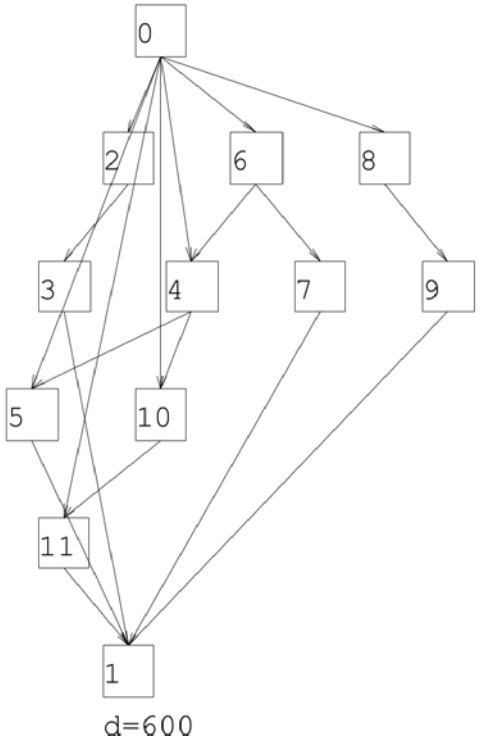


Outline

- Motivation
- Dynamic Partial Reconfiguration:
  - Background
  - Features
  - Aging mitigation
- **System model**
- System Design methodology
- Experiment and Results
- Conclusion

System model: *Application*

- **Task-graph**
- **Parameters for problem formulation**



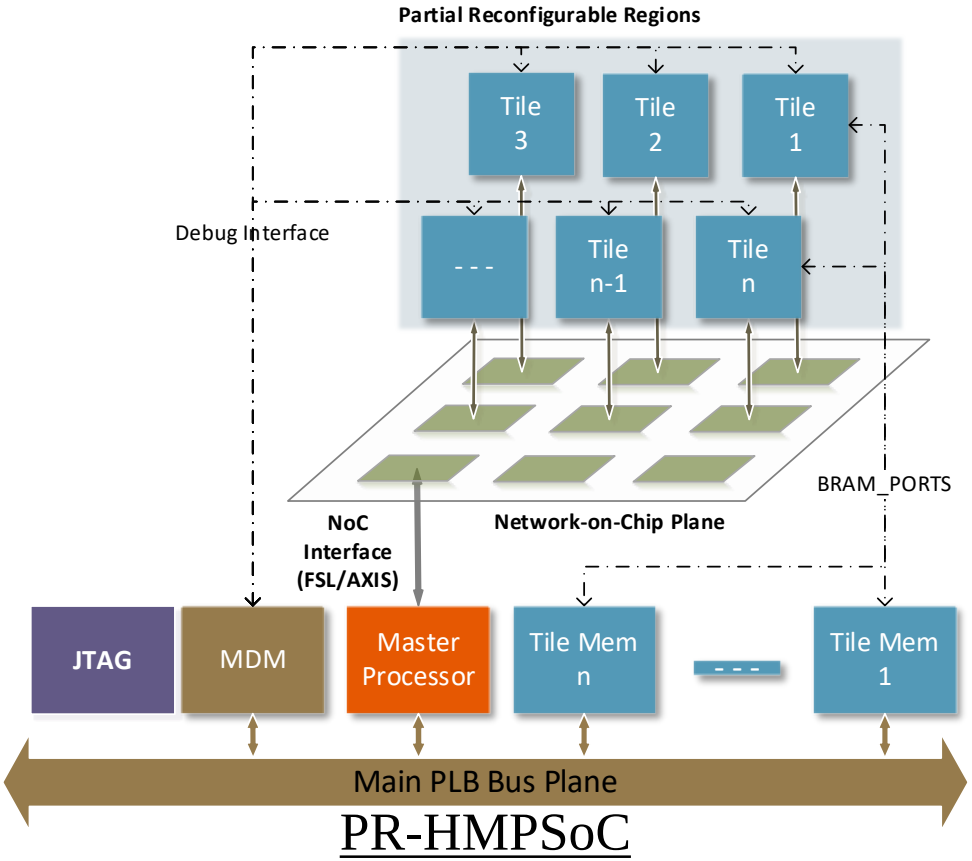
Application Task-graph

Parameter	Description
$TaskID$	Serial number of task
$TaskType$	Type of PRM used
$StartT$	Start time of task
$ExecT$	Expected execution time
$EndT$	End time of task
$TaskCLBs$	CLBs used for PRM implementation
$TaskBRAMs$	BRAMs used for PRM implementation
$TaskDSPs$	DSPs used for PRM implementation
$TaskMTTF$	Expected MTTF of the task PRM
$TaskD$	Any soft/hard deadline of the task

Task-level parameters

System model: *Architecture* [Nguyen2014]

- **PR-HMPSoC : PRRs and Static components**
- **NoC-based system**



PR-HMPSoC

Parameter	Description
$prrID_r$	Serial number of PRR
$prrCLBs_r$	CLBs present in the PRR
$prrBRAMs_r$	BRAMs present in the PRR
$prrDSPs_r$	DSPs present in the PRR
$prrMTTF_r$	Estimated MTTF of the PRR
$prrPRMs_r$	List of PRMs supported by the PRR
$prrExTrace_r$	Schedule of task execution on the PRR

PRR parameters

T. D. A. Nguyen and A. Kumar. PR-HMPSoC: A versatile partially reconfigurable heterogeneous Multiprocessor System-on-Chip for dynamic FPGA-based embedded systems. In *Proceedings of FPL*, 2014

System model: *Reliability* [Xiang2010]

Workload *Stress*

Hardware profile

$$R(t) = e^{-(t/\eta)^{\beta}}$$

$$MTTF = \eta \times \Gamma(1 + 1/\beta)$$

Lifetime Reliability

$$\eta_{eff} = \frac{\sum \Delta t_i}{\sum \frac{\Delta t_i}{\eta_i}} \qquad t = \sum \Delta t_i \qquad \eta_i = \frac{MTTF_i}{\Gamma(1 + 1/\beta)}$$

"Effective" stress

$$prrMTTF_r = \frac{P_{app}}{\sum_{i=1}^{M} \frac{ExecT_i}{TaskMTTF_i}}$$

$$SysMTTF = \min_{all\ PRRs}(prrMTTF_r)$$

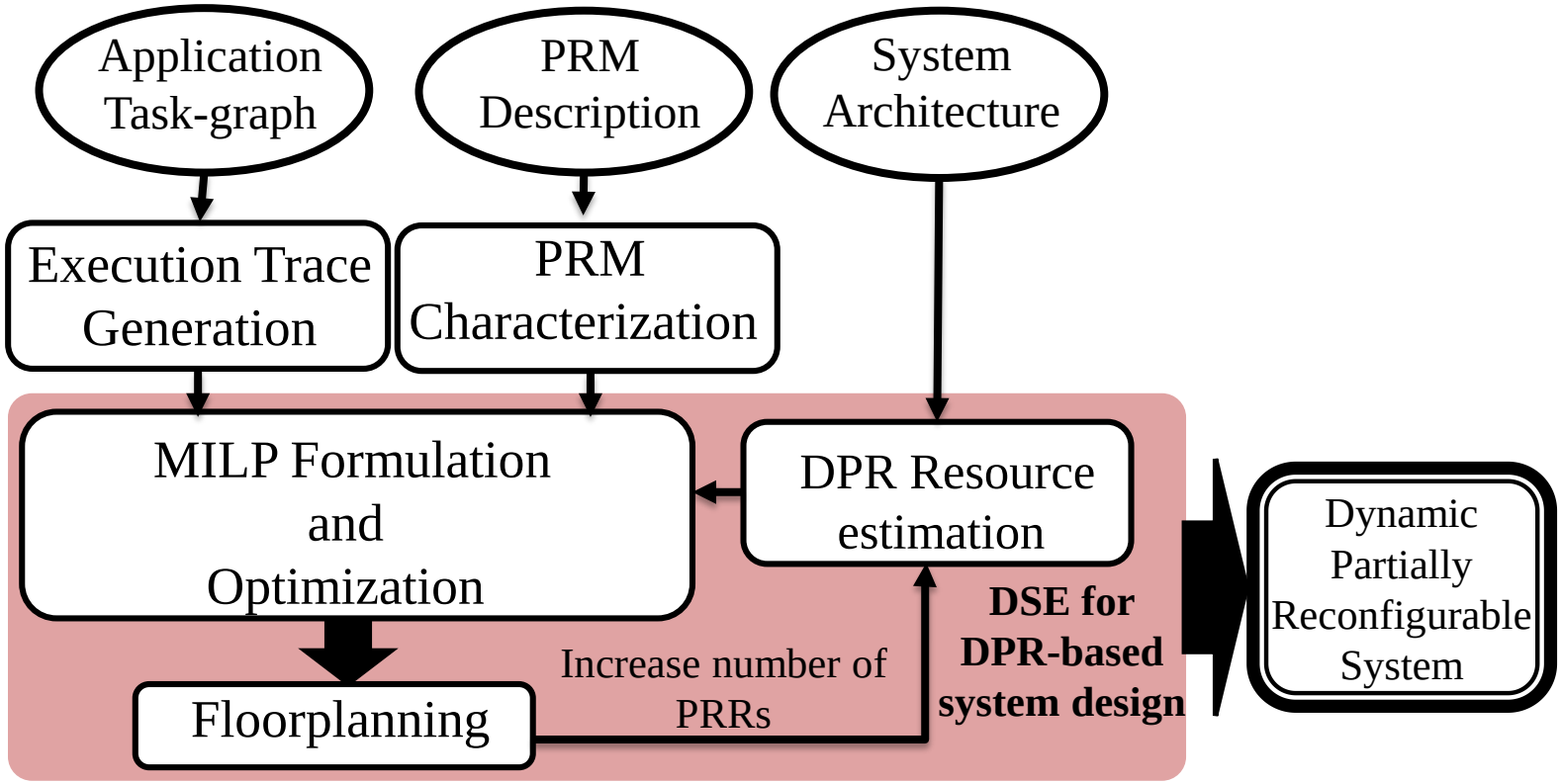
Expected time to next permanent fault

Y. Xiang, T. Chantem, R. P. Dick, X. S. Hu, and L. Shang, "System-level reliability modeling for MPSoCs," in *CODES*, 2010

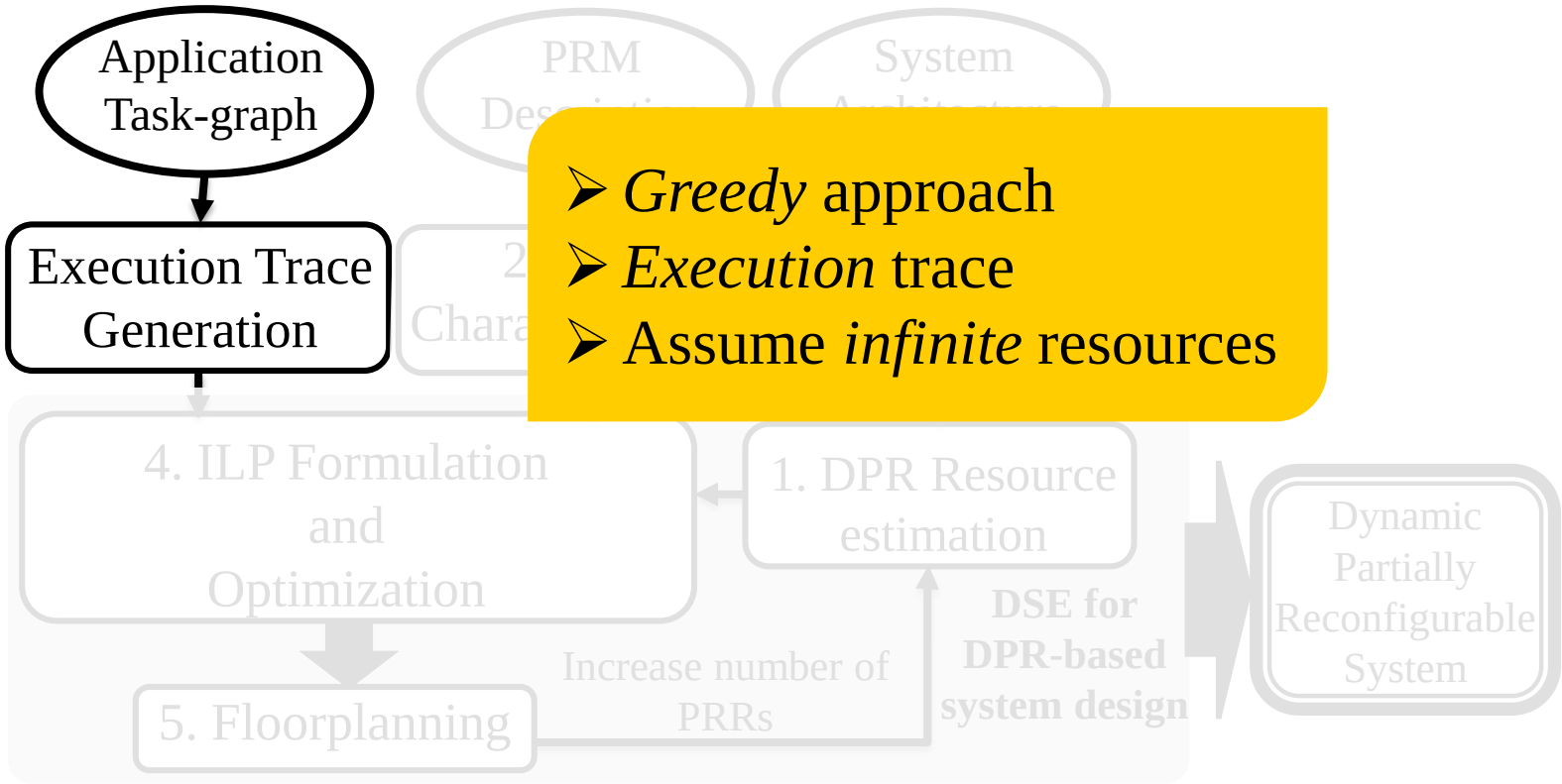
Outline

- Motivation
- Dynamic Partial Reconfiguration:
  - Background
  - Features
  - Aging mitigation
- System model
- **System Design methodology**
- Experiment and Results
- Conclusion

System Design Methodology

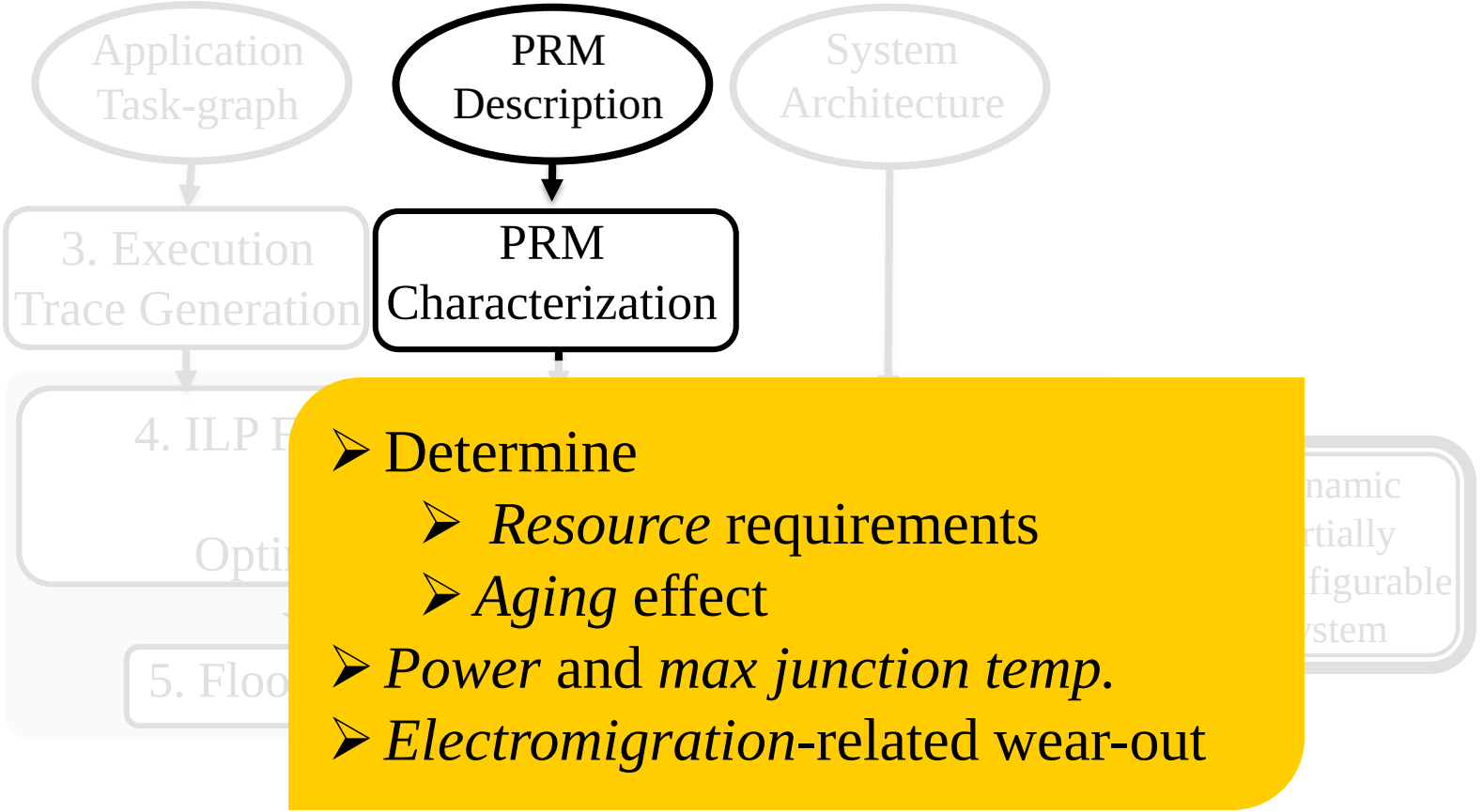


System Design Methodology



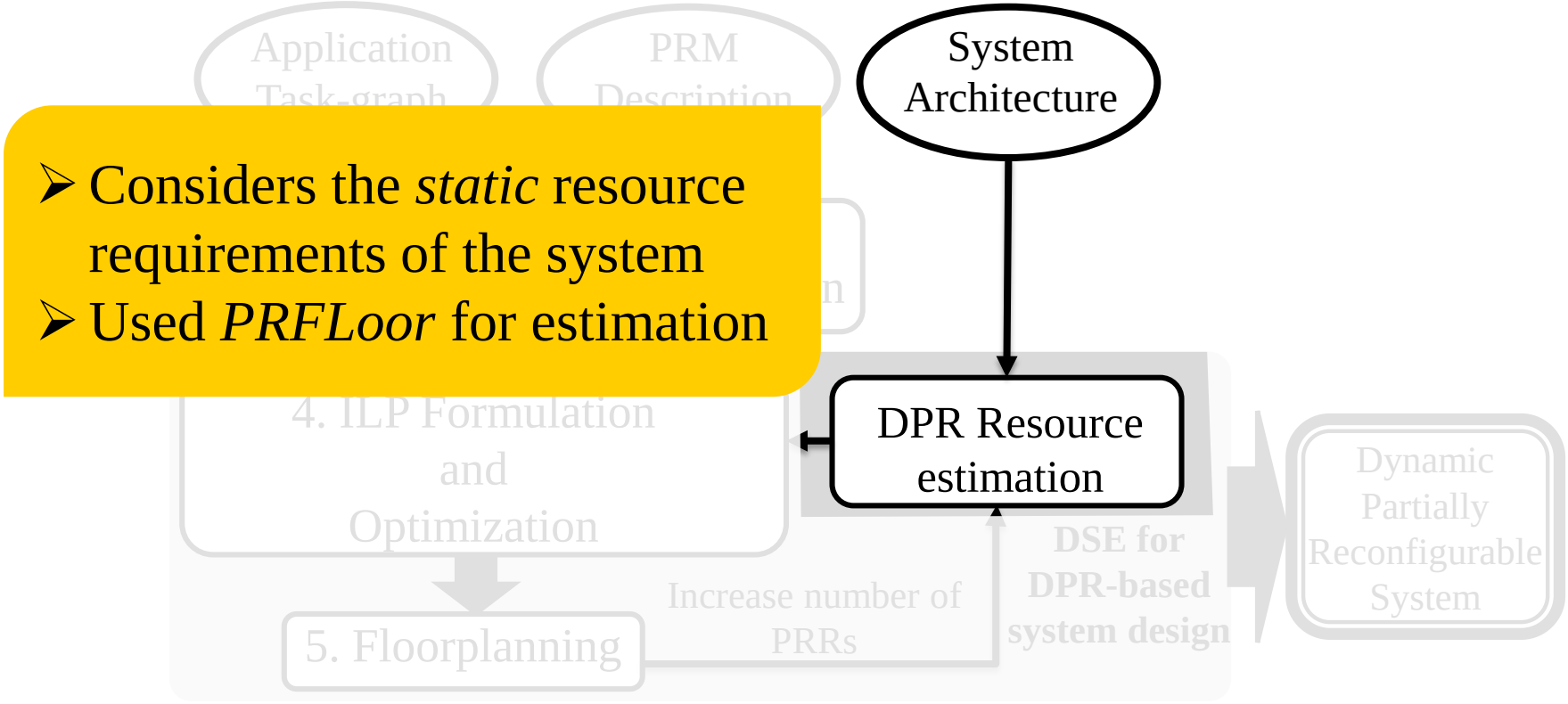
- *Greedy* approach
- *Execution* trace
- Assume *infinite* resources

System Design Methodology



- Determine
  - *Resource* requirements
  - *Aging* effect
- *Power* and *max junction temp.*
- *Electromigration*-related wear-out

System Design Methodology



- Considers the *static* resource requirements of the system
- Used *PRFLoor* for estimation

T. D. Nguyen and A. Kumar. PRFloor: An Automatic Floorplanner for Partially Reconfigurable FPGA Systems. In *Proceedings of FPGA*, 2016.

System Design Methodology

Exec_Trace / *PRRs*	T_1	T_2	.....	T_t	.....	T_T
R_1	$rt_{(1,1)}$	$rt_{(1,2)}$	....	$rt_{(1,t)}$	....	$rt_{(1,T)}$
R_2	$rt_{(2,1)}$	$rt_{(2,2)}$	....	$rt_{(2,t)}$	....	$rt_{(2,T)}$
:	:	:	..:...	:		:
R_r	$rt_{(r,1)}$	$rt_{(r,2)}$	....	$rt_{(r,t)}$	....	$rt_{(r,T)}$
:	:	:		:		:
R_R	$rt_{(R,1)}$	$rt_{(R,2)}$	....	$rt_{(R,t)}$	....	$rt_{(R,T)}$

- Objectives:
  - Minimize *makespan*
  - Maximize *SysMTTF*

3. Execution Trace Generation

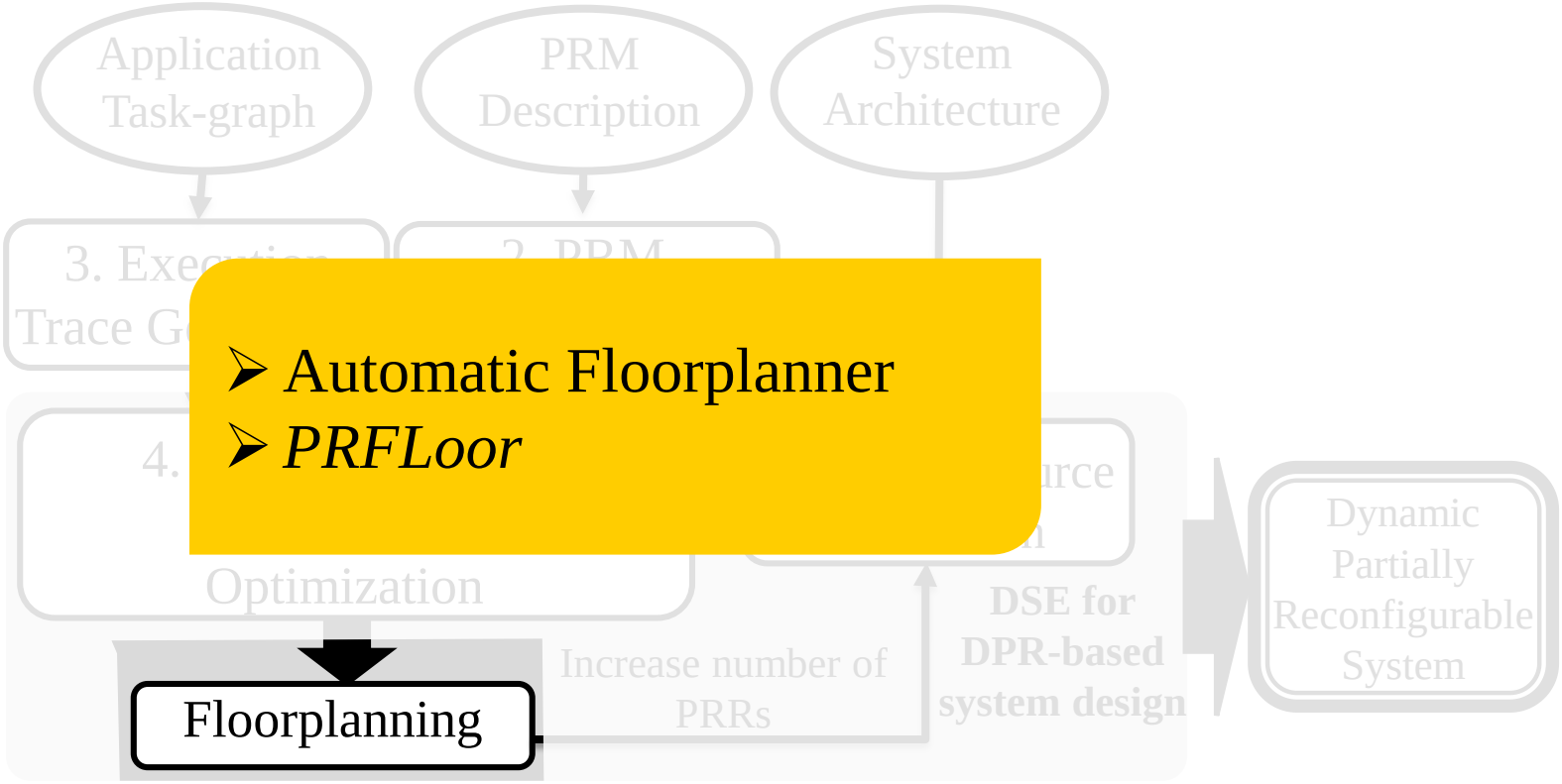
2. PRM Characterization

MILP Formulation and Optimization

5. Floorplanning

- Constraints:
  - *Deadline*
  - Task-level *precedence*
  - *PRR-level* MTTF
  - *Resource* constraints

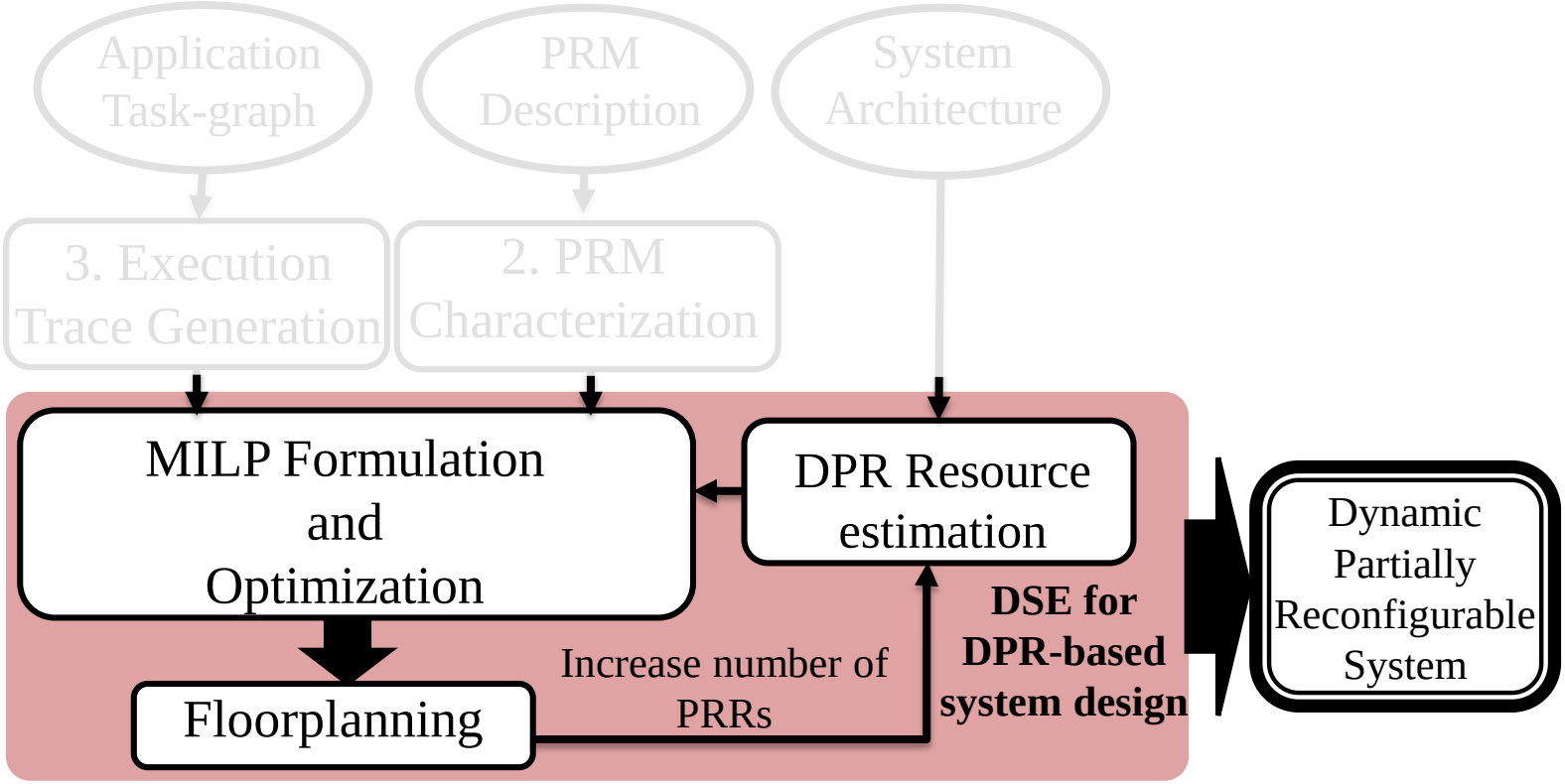
System Design Methodology



- Automatic Floorplanner
- *PRFLoor*

T. D. Nguyen and A. Kumar. PRFloor: An Automatic Floorplanner for Partially Reconfigurable FPGA Systems. In *FPGA*, 2016

System Design Methodology



Outline

- Motivation
- Dynamic Partial Reconfiguration:
  - Background
  - Features
  - Aging mitigation
- System model
- System Design methodology
- **Experiment and Results**
- Conclusion

Experiments and Results

Experiment Setup :

- **Two CPUs: Intel Xeon E5-2609 v2 @ 2.50GHz (quad-core), 32 GB of memory**
- **Ubuntu 14.04 LTS 64-bit**
- **Virtex-6 XC6VLX240T**
- ***Gurobi* Solver for finding MILP solution**
- **Task-graphs generated using *TGFF***

R. P. Dick, D. L. Rhodes, and W. Wolf. TGFF: task graphs for free. In *CODES*, pages 97–101. IEEE Computer Society, 1998
Gurobi. 2017. Gurobi Optimization version 6.0.2. www.gurobi.com. (2017).

Experiments and Results

Experiment Setup :

- **IP Pool:**
  - *50* real-world hardware accelerators
  - Synthesized using Xilinx Vivado Suite (ver 16.2)

PRM	LUT	BRAM	DSP	Source
DFDIV	7309	1	24	CHStone
DFMUL	4051	1	16	CHStone
Log2	8212	0	0	EPFL
ADPCM	6222	6	126	OpenCores
FFT1024	19796	18	52	OpenCores
SHA	3069	20	0	OpenCores
JPEG	6581	11	10	OpenCores
Video Stream Scaler	524	2	11	Xilinx
Video Test Pattern	2543	3	12	Xilinx
Microblaze (*Max Area*)	5539	5	6	Xilinx

Some notable PRMs used in experiments

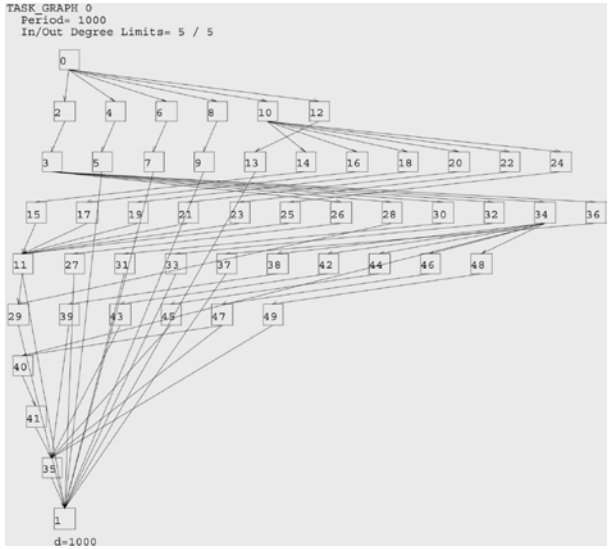
Xilinx. 2017. Intellectual Property. www.xilinx.com/products/intellectualproperty.html. (2017).
EPFL. 2017. Combinational Benchmark Suite. lsi.epfl.ch/benchmarks. (2017)
OpenCores. 2017. www.opencores.org. (2017)
Yuko Hara, Hiroyuki Tomiyama, Shinya Honda, and Hiroaki Takada. 2009. Proposal and quantitative analysis of the CHStone benchmark program suite for practical C-based high-level synthesis. *Journal of Information Processing* (2009).

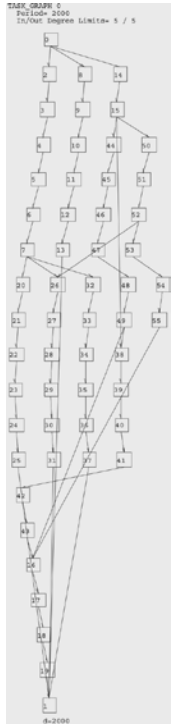
Experiments and Results

- **Experiment Setup :**

  Optimization modes :
  - ***homogeneous*** **/** ***heterogeneous***
  - **Minimize** ***makespan*** **/**
    Maximize system ***MTTF***

  Task-graph types :
  - **Parallelism:** ***Fat*** **/** ***Slim***

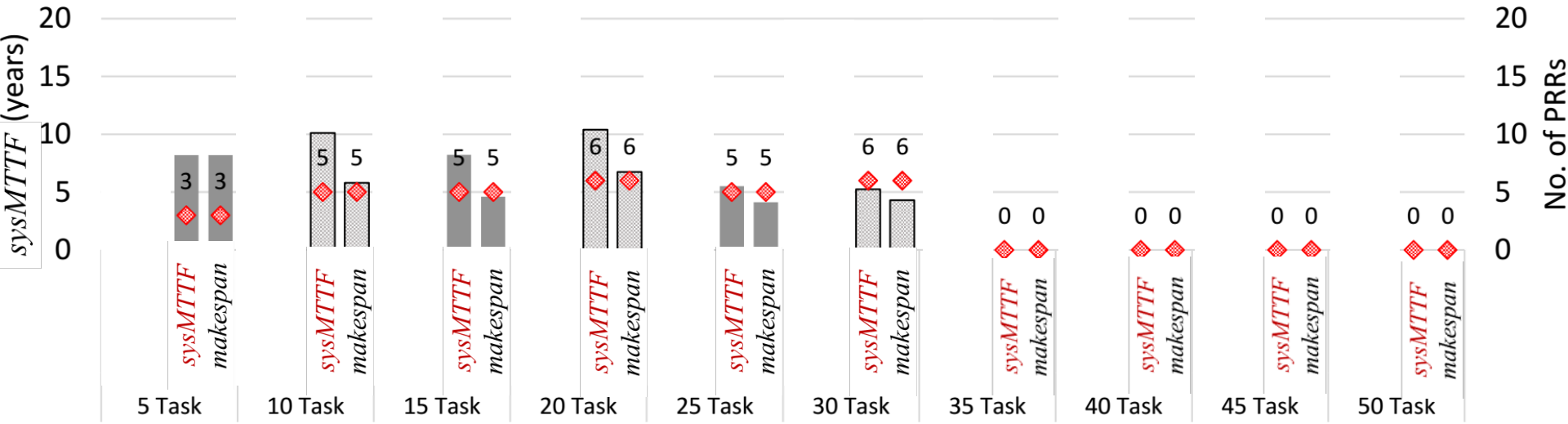




Experiments and Results

Results: *Fat* graphs

- System MTTF-aware scheduling : ***homogeneous*** PRRs

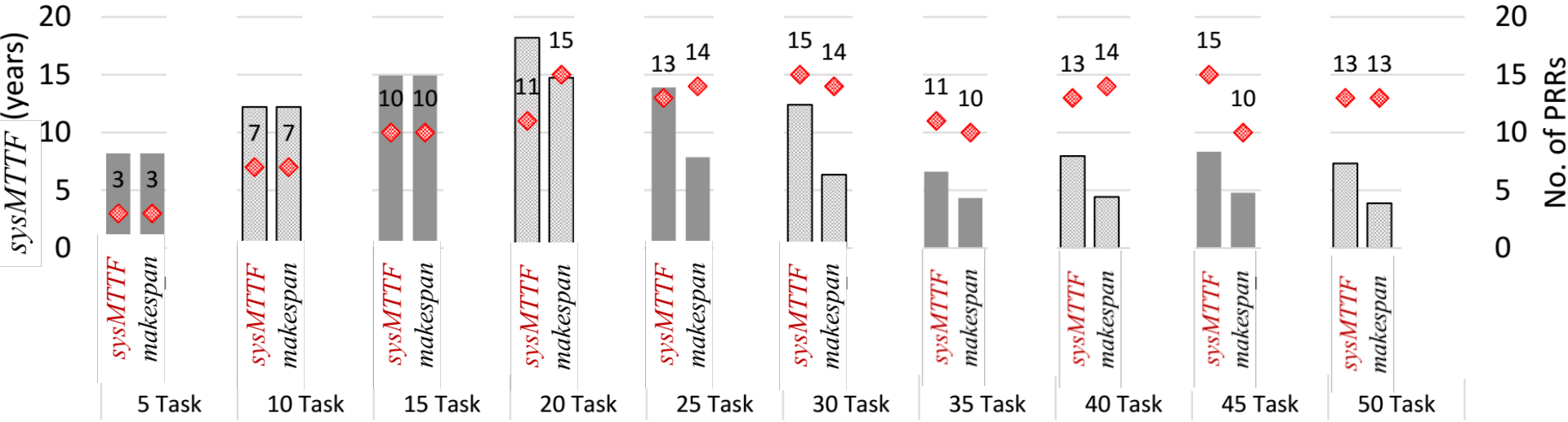


sysMTTF: Maximize system MTTF with deadline constraints
makespan: Minimize makespan with deadline constraints

Experiments and Results

Results: *Fat* graphs

- System MTTF-aware scheduling : ***heterogeneous*** PRRs

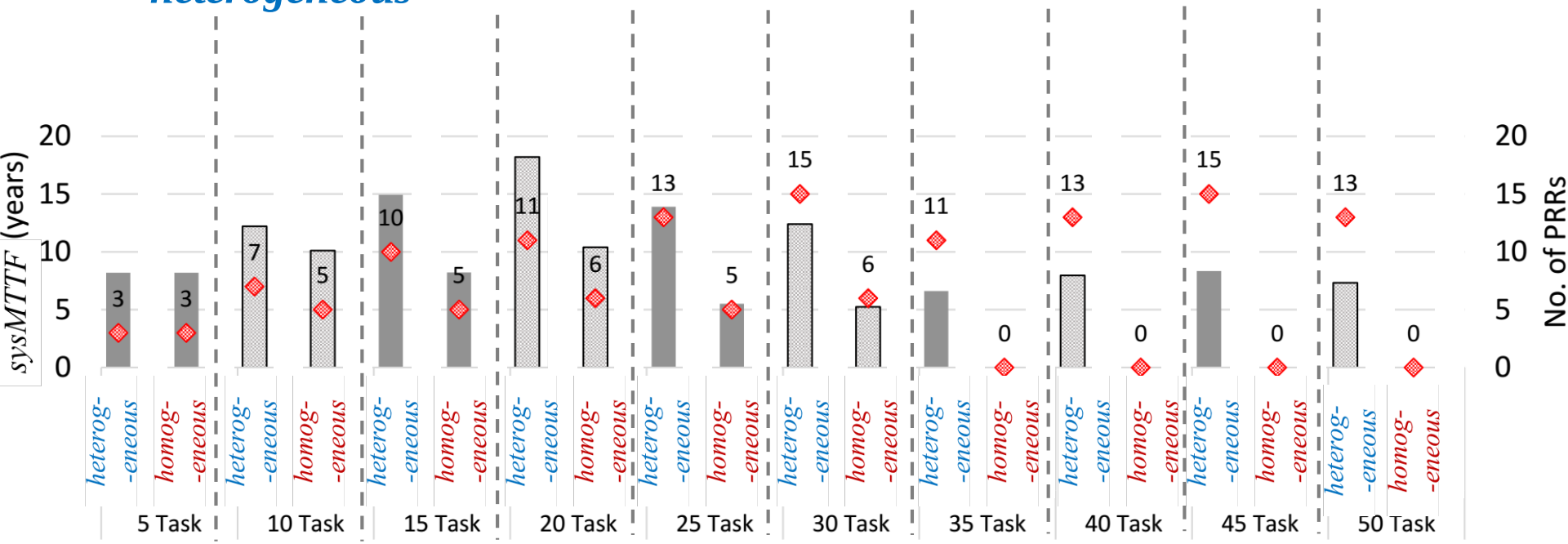


sysMTTF: Maximize system MTTF with deadline constraints
makespan: Minimize makespan with deadline constraints

Experiments and Results

Results: *Fat* graphs

- **System MTTF-aware system partitioning: *homogeneous* v/s *heterogeneous***



homog--eneous Maximize system MTTF with deadline constraints using *homogeneous* PRRs

heterog--eneous Maximize system MTTF with deadline constraints using *heterogeneous* PRRs

Experiments and Results

Results: *Summary*

Scenarios	T= 5	T=10	T=15	T=20	T=25	T=30	T=35	T=40	T=45	T=50
Fat, Large	0.00	0.21	0.82	0.75	1.52	1.37	**6.62**	**7.96**	**8.33**	**7.33**
Slim, Large	0.00	0.00	1.24	1.36	1.42	1.95	**9.57**	1.76	**13.16**	1.13

SysMTTF Improvements of Heterogeneous vs. Homogeneous Systems

$$\frac{sysMTTF_{hetero} - sysMTTF_{homo}}{sysMTTF_{homo}}$$

Experiments and Results

Results: *Summary*

Scenarios	T= 5	T=10	T=15	T=20	T=25	T=30	T=35	T=40	T=45	T=50
Fat, Large	0.00	0.21	0.82	0.75	1.52	1.37	**6.62**	**7.96**	**8.33**	**7.33**
Slim, Large	0.00	0.00	1.24	1.36	1.42	1.95	**9.57**	1.76	**13.16**	1.13
Fat, Small	0.00	0.00	0.00	0.00	0.00	0.00	0.17	0.06	0.06	**0.00**
Slim, Small	0.00	0.00	0.00	0.00	0.05	0.11	0.00	0.00	0.08	0.00

SysMTTF Improvements of Heterogeneous vs. Homogeneous Systems

$$\frac{sysMTTF_{hetero} - sysMTTF_{homo}}{sysMTTF_{homo}}$$

Conclusion

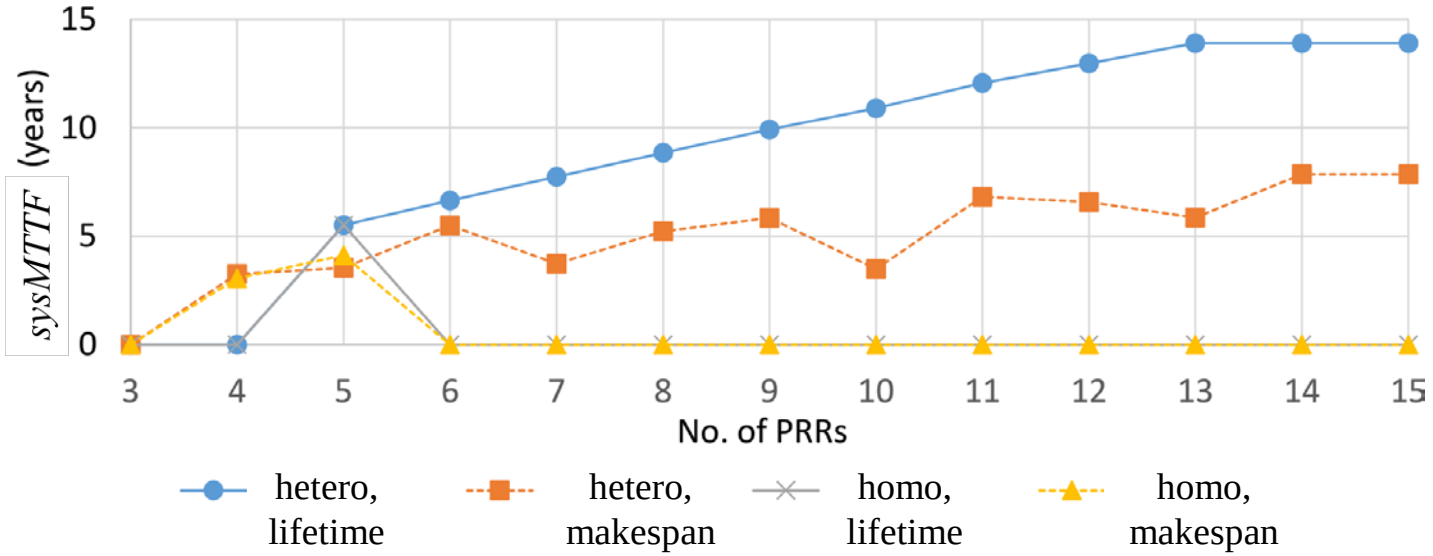
- A design methodology for lifetime-aware DPR-based systems was proposed
  - Scheduling with *aging-estimation*
  - Integration of *resource constraints* into scheduler
- Investigated *homogeneous* v/s *heterogeneous* PRRs
- Investigate trade-off between *aging-related* and *externally-induced* permanent faults (*future work*)
- Use other *global* optimization methods (*future work*)

Thank You

Queries ?

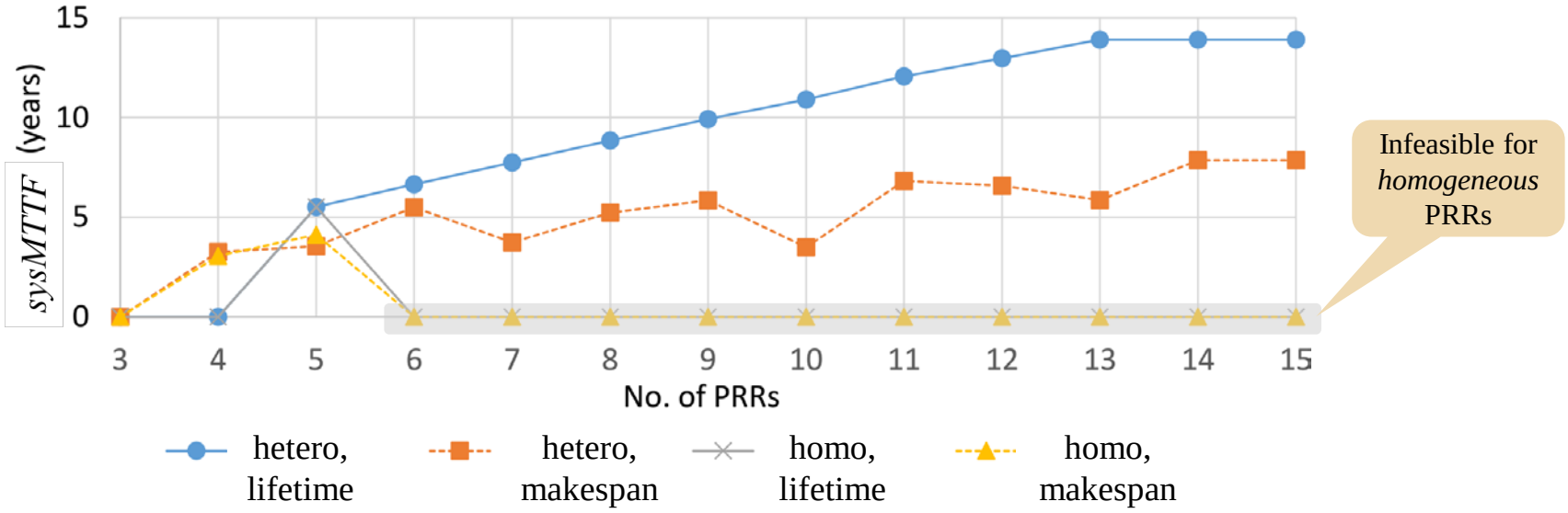
Experiments and Results

Results: Variation of System MTTF with #PRRs in a typical application with 25 tasks



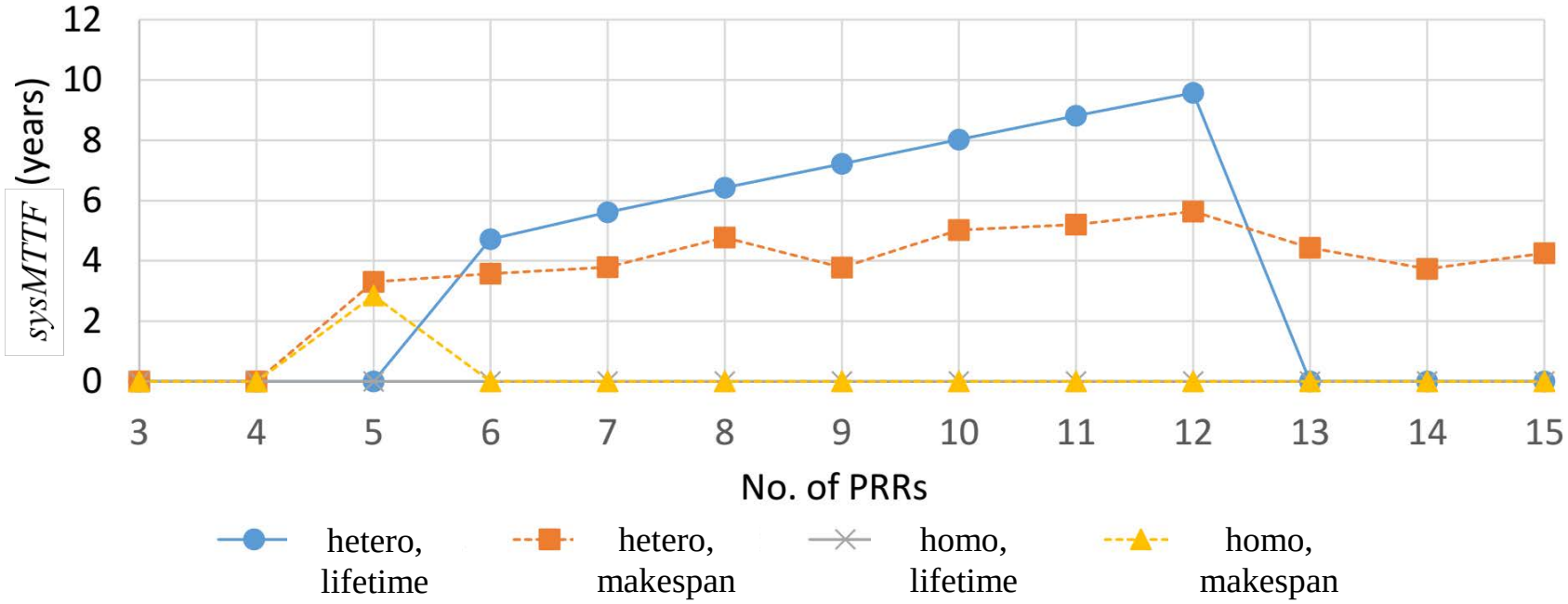
Experiments and Results

Results: Variation of System MTTF with #PRRs in a typical application with 25 tasks



Experiments and Results

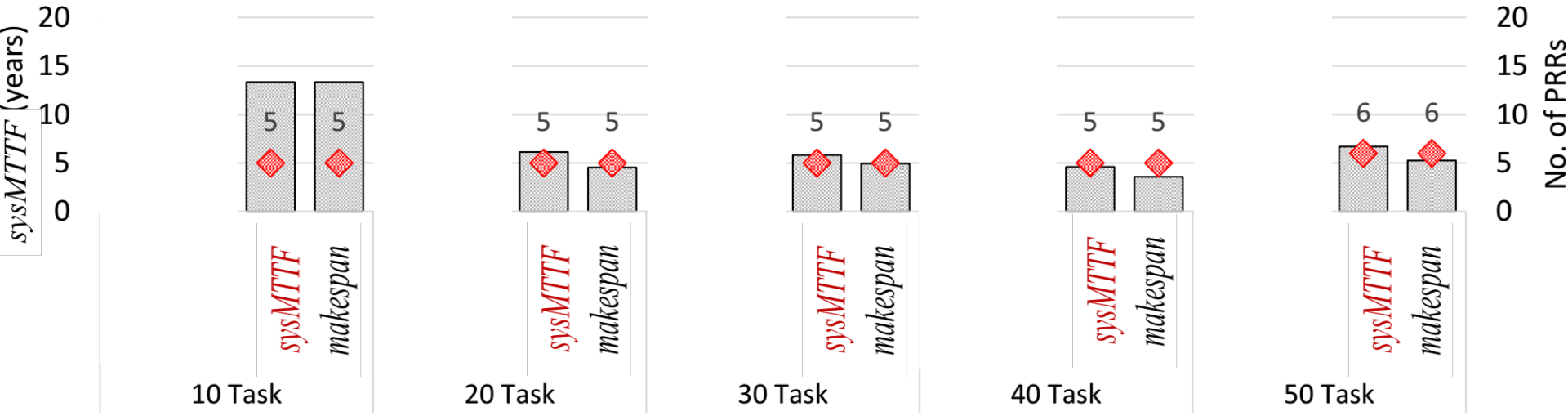
Results: *Slim* graphs



Experiments and Results

Results: *Slim* graphs

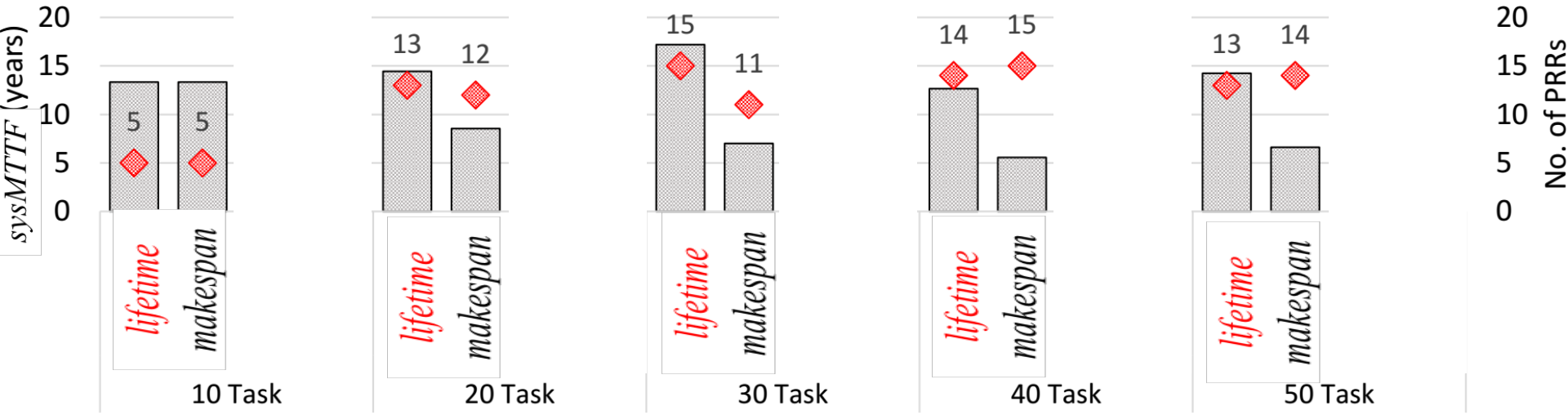
- System MTTF-aware scheduling : *homogeneous* PRRs



Experiments and Results

Results: *Slim* graphs

- System MTTF-aware scheduling : *heterogeneous* PRRs



Experiments and Results

Results: *Slim* graphs

- System MTTF-aware system partitioning: *homogeneous* v/s *heterogeneous*