Assigning Real-Time Tasks on Heterogeneous Multiprocessors with Two Unrelated Types of Processors

Björn Andersson, Gurulingesh Raravi and Konstantinos Bletsas

Real-time scheduling on a uniprocessor



time

Real-time scheduling Real-time scheduling on a uniprocessor on a multiprocessor







Scheduling related challenges for heterogeneous multiprocessors in real-time systems:

-Precedence constraints

-Sharing of low-level hardware resources (caches, interconnection networks);

-The execution time of a task depends on which processor it executes on.



Scheduling related challenges for heterogeneous multiprocessors in real-time systems:

-Precedence constraints

-Sharing of low-level hardware resources (caches, interconnection networks);

-The execution time of a task depends on which processor it executes on.

Focus of this talk.

Different views on a heterogeneous multiprocessors:

A heterogeneous multiprocessor is synthesized for a specific application.

A heterogeneous multiprocessor is a general-purpose computing platform.

Different views on a heterogeneous multiprocessors:

A heterogeneous multiprocessor is a general-purpose computing platform.

View taken in this talk.

How many different types of processors does the computer system have?

-Two types of processors

- More that two types of processors

$$- P P P P P P P P$$

How many different types of processors does the computer system have?



Different assumptions about task migration:

- -A task can migrate to any processor;
- -A task can migrate but only between processors of the same type;
- -A task cannot migrate.

Different assumptions about task migration:

- -A task can migrate to any processor;
- -A task can migrate but only between processors of the same type;

-A task cannot migrate.

Considered in this talk.

-Dependent tasks: An arrival of a task is dependent on an event related to another task.

- Independent tasks: An arrival of a task is independent of events related to other tasks.
 - + periodic tasks
 - * implicit deadline
 - * explicit deadline
 - + sporadic tasks
 - * implicit deadline
 - * explicit deadline

-Dependent tasks: An arrival of a task is dependent on an event related to another task.

- Independent tasks: An arrival of a task is independent of events related to other tasks.
 - + periodic tasks
 - * implicit deadline
 - * explicit deadline
 - + sporadic tasks
 - * implicit deadline
 - * explicit deadline

Considered in this talk.

- Dependent tasks: An arrival of a task is dependent on an event related to another task.
- Independent tasks: An arrival of a task is independent of events related to other tasks.
 - + periodic tasks
 - * implicit deadline
 - * explicit deadline
 - + sporadic tasks
 - * implicit deadline
 - * explicit deadline

Different scheduling algorithms:

- RM

- EDF

- Dependent tasks: An arrival of a task is dependent on an event related to another task.
- Independent tasks: An arrival of a task is independent of events related to other tasks.
 - + periodic tasks
 - * implicit deadline
 - * explicit deadline
 - + sporadic tasks

implicit deadline
explicit deadline

Different scheduling algorithms:



- EDF

Considered in this talk.

Model

- *P*¹ denotes the set of all processors of type-1.
- *P*² denotes the set of all processors of type-2.
- τ denotes a set of tasks $\tau = \{\tau_1, \tau_2, \dots, \tau_n\};$
- A task τ_i assigned to a processor of type-1 has utilization U_i^1 .
- A task τ_i assigned to a processor of type-2 has utilization U_i^2 .

Problem statement

Assign tasks to processors so that each processor is utilized to at most 100%.

Example of a problem instance

$$\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{\mathsf{P}_1\}, P^2 = \{\mathsf{P}_2, \mathsf{P}_3\}.$$



	Processor type-1	Processor type-2
$ au_1$	$U_1^{1}=0.90$	$U_1^2 = 0.40$
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	<i>U</i> ₄ ² =0.80



Example of a problem instance

$\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} \ P^1 = \{\mathsf{P}_1\}, \ P^2 = \{\mathsf{P}_2, \mathsf{P}_3\}.$

	Processor type-1	Processor type-2	
$ au_1$	$U_1^{1}=0.90$	$U_1^2 = 0.40$	
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$	τ_4 τ_2
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$	
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$	P_2 P_3

We can do the assignment like this.

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	$U_1^{1}=0.90$	$U_1^2 = 0.40$
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	<i>U</i> ₄ ² =0.80



 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	$U_1^{1}=0.90$	$U_1^2 = 0.40$
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



21

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	$U_1^1 = 0.90$	$U_1^2 = 0.40$
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	<i>U</i> ¹ =0.90	$U_1^2 = 0.40$
$ au_2$	<i>U</i> ₂ ¹ =0.90	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	<i>U</i> ¹ =0.90	$U_1^2 = 0.40$
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$
$ au_3$	$U_3^1 = 0.40$	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	<i>U</i> ₄ ² =0.80



There is no processor on which τ_4 can be assigned.

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	<i>U</i> ¹ =0.90	$U_1^2 = 0.40$
$ au_2$	<i>U</i> ₂ ¹ =0.90	$U_2^2 = 0.40$
$ au_3$	$U_3^1 = 0.40$	$U_3^2 = 0.80$
$ au_4$	$U_4^{1}=0.40$	<i>U</i> ₄ ² =0.80



First-Fit fails on this task set.

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



$ au_1 \qquad U$	<i>V</i> ₁ ¹ =0.90	$U_1^2 = 0.40$
$ au_2$ U	<i>I</i> ₂ ¹ =0.90	$U_2^2 = 0.40$
$ au_3$ U	/ ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$ U	<i>I</i> ₄ ¹ =0.40	<i>U</i> ₄ ² =0.80



First-Fit has inifinite competitive ratio on heterogeneous multiprocessor with two types (shown in the paper)₂₆

Design Ideas

- Idea1: Try to assign a task on the processor where its utilization is smaller.
- Idea 2: if $U_i^1 \leq$ THRESHOLD and $U_i^2 >$ THRESHOLD then assign task τ_i to processor of type-1.

Design Ideas Partition the task set

- Idea1: Try to assign a task on the processor where its utilization is smaller.
- Idea 2: if $U_i^1 \leq$ THRESHOLD and $U_i^2 >$ THRESHOLD then assign task τ_i to processor of type-1.

 $\tau^{1}=\{\tau_{i} \propto \tau \text{ such that } U_{i}^{1} \leq U_{i}^{2}\}$ $\tau^{2}=\{\tau_{i} \propto \tau \text{ such that } U_{i}^{1} > U_{i}^{2}\}$

Design Ideas Partition the task set

- Idea1: Try to assign a task on the processor where its utilization is smaller.
- Idea 2: if $U_i^1 \leq$ THRESHOLD and $U_i^2 >$ THRESHOLD then assign task τ_i to processor of type-1.

 $\tau^{1} = \{ \tau_{i} \propto \tau \text{ such that } U_{i}^{1} \leq U_{i}^{2} \}$ $\tau^{2} = \{ \tau_{i} \propto \tau \text{ such that } U_{i}^{1} > U_{i}^{2} \}$

 $H1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 > 1/2 \}$ $F1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 \le 1/2 \}$

Design Ideas Partition the task set

- Idea1: Try to assign a task on the processor where its utilization is smaller.
- Idea 2: if $U_i^1 \leq$ THRESHOLD and $U_i^2 >$ THRESHOLD then assign task τ_i to processor of type-1.

 $\tau^1 = \{ \tau_i \propto \tau \text{ such that } U_i^1 \leq U_i^2 \}$ $\tau^2 = \{ \tau_i \propto \tau \text{ such that } U_i^1 > U_i^2 \}$

 $H1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 > 1/2 \}$ $F1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 \le 1/2 \}$ $H2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 > 1/2 \}$ $F2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 \le 1/2 \}$

Algorithm Outline Partition the task set

- 1. Form the sets H1, H2, F1, F2
- 2. first-fit(H1, *P*¹)
- 3. first-fit(H2, P²)
- 4. first-fit(F1, P^1)
- 5. first-fit(F2, P²)

 $\tau^1 = \{ \tau_i \propto \tau \text{ such that } U_i^1 \leq U_i^2 \}$ $\tau^2 = \{ \tau_i \propto \tau \text{ such that } U_i^1 > U_i^2 \}$

 $H1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 > 1/2 \}$ $F1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 \le 1/2 \}$ $H2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 > 1/2 \}$ $F2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 \le 1/2 \}$

Algorithm Partition the task set

- 1. Form sets H1, H2, F1, F2
- 2. $\forall p: U[p] := 0$
- 3. $\forall p: \tau[p] := \emptyset$
- 4. if first-fit(H1, P^1) $\neq H1$ then declare FAILURE
- 5. if first-fit(H2, P^2) $\neq H2$ then declare FAILURE
- 6. F11 :=first-fit($F1, P^1$)
- 7. F22 :=first-fit($F2, P^2$)
- 8. if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS
- 9. if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE
- 10. if $(F11 \neq F1) \land (F22 = F2)$ then
- 11. $F12 := F1 \setminus F11$
- 12. **if** first-fit($F12, P^2$) = F12 then
- 13. declare SUCCESS
- 14. else
- 15. declare FAILURE
- 16. end
- 17. end
- 18. if $(F11 = F1) \land (F22 \neq F2)$ then
- 19. $F21 := F2 \setminus F22$
- 20. if first-fit(F21, P^1) = F21 then
- 21. declare SUCCESS
- 22. else
- 23. declare FAILURE
- 24. end
- 25. end

- $\tau^1 = \{ \tau_i \propto \tau \text{ such that } U_i^1 \leq U_i^2 \}$ $\tau^2 = \{ \tau_i \propto \tau \text{ such that } U_i^1 > U_i^2 \}$
- $H1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 > 1/2 \}$ $F1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 \le 1/2 \}$ $H2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 > 1/2 \}$ $F2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 \le 1/2 \}$

<u>FF-3C</u>

Partition the task set

- 1. Form sets H1, H2, F1, F2
- 2. $\forall p: U[p] := 0$
- 3. $\forall p: \tau[p] := \emptyset$
- 4. if first-fit($H1, P^1$) $\neq H1$ then declare FAILURE
- 5. if first-fit(H2, P^2) $\neq H2$ then declare FAILURE
- 6. F11 :=first-fit($F1, P^1$)
- 7. F22 :=first-fit($F2, P^2$)
- 8. if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS
- 9. if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE
- 10. if $(F11 \neq F1) \land (F22 = F2)$ then
- 11. $F12 := F1 \setminus F11$
- 12. **if** first-fit($F12, P^2$) = F12 then
- 13. declare SUCCESS
- 14. else
- 15. declare FAILURE
- 16. end
- 17. end
- 18. if $(F11 = F1) \land (F22 \neq F2)$ then
- 19. $F21 := F2 \setminus F22$
- 20. if first-fit(F21, P^1) = F21 then
- declare SUCCESS
- 22. else
- 23. declare FAILURE
- 24. end
- 25. end

- $\tau^1 = \{ \tau_i \propto \tau \text{ such that } U_i^1 \leq U_i^2 \}$ $\tau^2 = \{ \tau_i \propto \tau \text{ such that } U_i^1 > U_i^2 \}$
- $H1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 > 1/2 \}$ $F1 = \{ \tau_i \propto \tau^1 \text{ such that } U_i^2 \le 1/2 \}$ $H2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 > 1/2 \}$ $F2 = \{ \tau_i \propto \tau^2 \text{ such that } U_i^1 \le 1/2 \}$

<u>FF-3C</u>

- 1. Form sets H1, H2, F1, F2
- 2. $\forall p: U[p] := 0$
- 3. $\forall p: \tau[p] := \emptyset$
- 4. if first-fit(H1, P^1) $\neq H1$ then declare FAILURE
- 5. if first-fit(H2, P^2) $\neq H2$ then declare FAILURE
- 6. F11 :=first-fit($F1, P^1$)
- 7. F22 :=first-fit($F2, P^2$)
- 8. if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS
- 9. if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE
- 10. if $(F11 \neq F1) \land (F22 = F2)$ then
- 11. $F12 := F1 \setminus F11$
- 12. **if** first-fit($F12, P^2$) = F12 then
- declare SUCCESS
- 14. else
- declare FAILURE
- 16. end
- 17. end
- 18. if $(F11 = F1) \land (F22 \neq F2)$ then
- 19. $F21 := F2 \setminus F22$
- 20. if first-fit(F21, P^1) = F21 then
- declare SUCCESS
- 22. else
- 23. declare FAILURE
- 24. end 25. end

function first-fit(ts : set of tasks; ps : set of processors) 1. return set of tasks 2. assigned tasks := \emptyset 3 If ps consists of type-1 (type-2) processors, then order ts by decreasing U_i^2/U_i^1 (resp., incr. U_i^1/U_i^2). Use any order for processors ps, but maintain it during the execution of the function first-fit. $\tau_i :=$ first task in ts 4. 5. p := first processor in ps 6. Let k denote the type of processor p (either 1 or 2) if U[p]+ $U_i^k \leq 1$ then 7. $U[p] := U[p] + U_i^k$ 8. 9. $\tau[\mathbf{p}] \coloneqq \tau[\mathbf{p}] \cup \{\tau_i\}$ 10. assigned_tasks := assigned_tasks $\cup \{\tau_i\}$ 11. if remaining tasks exist in ts then 12. $\tau_i := \text{next task in ts}$ 13. go to line 5. 14. else 15. return assigned_tasks 16. end if 17. else 18. if remaining processors exist in ps then 19.p := next processor in ps20.go to line 6. 21.else 22.return assigned_tasks 23.end if 24.end if

Applying FF-3C on an example

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



	Processor type-1	Processor type-2
$ au_1$	$U_1^{1}=0.90$	$U_1^2 = 0.40$
$ au_2$	$U_2^1 = 0.90$	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



<u>FF-3C</u>

		1.	function first-fit(ts : set of tasks; ps : set of processors)
1.	Form sets $H1$, $H2$, $F1$, $F2$	9	return set of tasks
2.		2. 3.	assigned_tasks := \emptyset If ps consists of type-1 (type-2) processors, then order
	$\forall p: \mathbf{U}[\mathbf{p}] := 0$	э.	ts by decreasing U_i^2/U_i^1 (resp., incr. U_i^1/U_i^2).
3.	$\forall p: \tau[p] := \emptyset$		Use any order for processors ps, but maintain it
4.	if first-fit($H1$, P^1) $\neq H1$ then declare FAILURE		during the execution of the function first-fit.
5.	if first-fit($H2$, P^2) $\neq H2$ then declare FAILURE	4.	$\tau_i :=$ first task in ts
6.	$F11 := $ first-fit($F1, P^1$)	5.	p := first processor in ps
7.	$F22 := \text{first-fit}(F2, P^2)$	6.	Let k denote the type of processor p (either 1 or 2)
8.	if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS	7.	if $U[p]+U_i^k \leq 1$ then
	if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE	8.	$\mathbf{U}[\mathbf{p}] := \mathbf{U}[\mathbf{p}] + U_i^k$
	if $(F11 \neq F1) \land (F22 = F2)$ then	9.	$\tau[\mathbf{p}] \coloneqq \tau[\mathbf{p}] \cup \{\tau_i\}$
11.	$F_{12} := F_1 \setminus F_{11}$	10.	assigned_tasks := assigned_tasks $\cup \{\tau_i\}$
12.	if first-fit($F12$, P^2) = $F12$ then	11. 12.	if remaining tasks exist in ts then $\tau_i := \text{next task in ts}$
		12.	τ_i = next task in ts go to line 5.
13.	declare SUCCESS	14.	else
14.	else	15.	return assigned_tasks
15.	declare FAILURE	16.	end if
16.	end	17.	else
17.	end	18.	if remaining processors exist in ps then
18.	if $(F11 = F1) \land (F22 \neq F2)$ then	19.	p := next processor in ps
19.	$F21 := F2 \setminus F22$	20.	go to line 6.
20.			
21.	Let us exect	it c	this line
22.		JIC	
23.	declate PAILUKE		
24.	end		36
25.	end		
Applying FF-3C on an example

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}. \quad (\tau_1)$



	Processor type-1	Processor type-2
τ_1	$U_1^{1} = 0.90$	$U_1^2 = 0.40$
τ_2	<i>U</i> ₂ ¹ =0.90	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



$$\tau^{1} = \{\tau_{3}, \tau_{4}\} \quad H1 = \{\tau_{3}, \tau_{4}\} \quad F1 = \{\}$$

$$\tau^{2} = \{\tau_{1}, \tau_{2}\} \quad H2 = \{\tau_{1}, \tau_{2}\} \quad F2 = \{\}$$

function first-fit(ts : set of tasks; ps : set of processors) 1. return set of tasks 1. Form sets H1, H2, F1, F22. assigned_tasks := \emptyset 2. $\forall p: U[p] := 0$ 3 If ps consists of type-1 (type-2) processors, then order ts by decreasing U_i^2/U_i^1 (resp., incr. U_i^1/U_i^2). 3. $\forall p: \tau[p] := \emptyset$ Use any order for processors ps, but maintain it 4. **if** first-fit($H1, P^1$) $\neq H1$ then declare FAILURE during the execution of the function first-fit. if first-fit($H2, P^2$) $\neq H2$ then declare FAILURE 5. $\tau_i :=$ first task in ts 4. F11 :=first-fit($F1, P^1$) 6. 5. p := first processor in ps 7. $F22 := \text{first-fit}(F2, P^2)$ 6. Let k denote the type of processor p (either 1 or 2) if U[p]+ $U_i^k \leq 1$ then 7. 8. if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS $U[p] := U[p] + U_i^k$ 8. 9. if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE 9. $\tau[\mathbf{p}] \coloneqq \tau[\mathbf{p}] \cup \{\tau_i\}$ 10. if $(F11 \neq F1) \land (F22 = F2)$ then 10. assigned_tasks := assigned_tasks $\cup \{\tau_i\}$ 11. $F12 := F1 \setminus F11$ if remaining tasks exist in ts then 11. if first-fit(F12, P^2) = F12 then 12.12. $\tau_i := \text{next task in ts}$ 13. go to line 5. 13.declare SUCCESS 14. else 14. else 15. return assigned_tasks 15.declare FAILURE 16. end if 16. end 17. else 17. end 18. if remaining processors exist in ps then 19. 18. if $(F11 = F1) \land (F22 \neq F2)$ then p := next processor in ps20.go to line 6. 19. $F21 := F2 \setminus F22$ 20.Let us execute this line. 21.22.23.deciale FAILORE 38 24.end 25. end

Applying FF-3C on an example

 $\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{P_1\}, P^2 = \{P_2, P_3\}.$



τ_1 $U_1^1=0.90$ $U_1^2=0.40$ τ_2 $U_2^1=0.90$ $U_2^2=0.40$ τ_3 $U_3^1=0.40$ $U_3^2=0.80$ τ_4 $U_4^1=0.40$ $U_4^2=0.80$		Processor type-1	Processor type-2
$\tau_3 = U_3^{1} = 0.40$ $U_3^{2} = 0.80$	$ au_1$	<i>U</i> ¹ =0.90	$U_1^2 = 0.40$
	τ_2	<i>U</i> ₂ ¹ =0.90	$U_2^2 = 0.40$
$\tau_{4} = U_{4}^{1} = 0.40$ $U_{4}^{2} = 0.80$	$ au_3$	$U_3^1 = 0.40$	$U_3^2 = 0.80$
4 4 4	$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



0.80

 $\tau^{1} = \{\tau_{3}, \tau_{4}\}$ H1= $\{\tau_{3}, \tau_{4}\}$ F1= $\{\}$ $\tau^2 = \{\tau_1, \tau_2\} H2 = \{\tau_1, \tau_2\} F2 = \{\}$

function first-fit(ts : set of tasks; ps : set of processors) 1. return set of tasks 1. Form sets H1, H2, F1, F22. assigned_tasks := \emptyset 2. $\forall p: U[p] := 0$ 3 If ps consists of type-1 (type-2) processors, then order ts by decreasing U_i^2/U_i^1 (resp., incr. U_i^1/U_i^2). 3. $\forall p: \tau[p] := \emptyset$ Use any order for processors ps, but maintain it 4. if first-fit($H1, P^1$) $\neq H1$ then declare FAILURE during the execution of the function first-fit. if first-fit($H2, P^2$) $\neq H2$ then declare FAILURE 5. $\tau_i :=$ first task in ts 4. $F11 := \text{first-fit}(F1, P^1)$ 6. 5. p := first processor in ps 6. Let k denote the type of processor p (either 1 or 2) 7. $F22 := \text{first-fit}(F2, P^2)$ if U[p]+ $U_i^k \leq 1$ then 7. 8. if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS $U[p] := U[p] + U_i^k$ 8. 9. if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE 9. $\tau[\mathbf{p}] \coloneqq \tau[\mathbf{p}] \cup \{\tau_i\}$ 10. if $(F11 \neq F1) \land (F22 = F2)$ then 10. assigned_tasks := assigned_tasks $\cup \{\tau_i\}$ 11. $F12 := F1 \setminus F11$ 11. if remaining tasks exist in ts then if first-fit(F12, P^2) = F12 then 12.12. $\tau_i := \text{next task in ts}$ 13. go to line 5. 13.declare SUCCESS 14. else 14. else 15. return assigned_tasks 15.declare FAILURE 16. end if 16. end 17. else 17. end 18. if remaining processors exist in ps then 19. 18. if $(F11 = F1) \land (F22 \neq F2)$ then p := next processor in ps20.go to line 6. 19. $F21 := F2 \setminus F22$ 20.Let us execute this line. 21.22.23.deciale FAILORE 40 24.end 25. end

Applying FF-3C on an example

$\tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} P^1 = \{\mathsf{P}_1\}, P^2 = \{\mathsf{P}_2, \mathsf{P}_3\}.$

	Processor type-1	Processor type-2
τ_1	<i>U</i> ¹ =0.90	$U_1^2 = 0.40$
$ au_2$	<i>U</i> ₂ ¹ =0.90	$U_2^2 = 0.40$
$ au_3$	<i>U</i> ₃ ¹ =0.40	$U_3^2 = 0.80$
$ au_4$	<i>U</i> ₄ ¹ =0.40	$U_4^2 = 0.80$



$$\tau^{1} = \{\tau_{3}, \tau_{4}\} \quad H1 = \{\tau_{3}, \tau_{4}\} \quad F1 = \{\}$$

$$\tau^{2} = \{\tau_{1}, \tau_{2}\} \quad H2 = \{\tau_{1}, \tau_{2}\} \quad F2 = \{\}$$

function first-fit(ts : set of tasks; ps : set of processors) return set of tasks 1. Form sets H1, H2, F1, F22. assigned_tasks := \emptyset 3 If ps consists of type-1 (type-2) processors, then order 2. $\forall p: U[p] := 0$ ts by decreasing U_i^2/U_i^1 (resp., incr. U_i^1/U_i^2). 3. $\forall p: \tau[p] := \emptyset$ Use any order for processors ps, but maintain it 4. if first-fit(H1, P^1) $\neq H1$ then declare FAILURE during the execution of the function first-fit. 5. if first-fit(H2, P^2) $\neq H2$ then declare FAILURE $\tau_i :=$ first task in ts 4. $F11 := \text{first-fit}(F1, P^1)$ 6. 5. p := first processor in ps F22 :=first-fit($F2, P^2$) 6. Let k denote the type of processor p (either 1 or 2) 7. if U[p]+ $U_i^k \leq 1$ then 7. 8. if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS $U[p] := U[p] + U_i^k$ 8. 9. if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE 9. $\tau[\mathbf{p}] \coloneqq \tau[\mathbf{p}] \cup \{\tau_i\}$ 10. if $(F11 \neq F1) \land (F22 = F2)$ then 10. assigned_tasks := assigned_tasks $\cup \{\tau_i\}$ 11. $F12 := F1 \setminus F11$ if remaining tasks exist in ts then 11. if first-fit(F12, P^2) = F12 then 12.12. $\tau_i := \text{next task in ts}$ 13. go to line 5. 13.declare SUCCESS 14. else 14. else 15. return assigned_tasks declare FAILURE 15.16. end if 16. end 17. else 17. end 18. if remaining processors exist in ps then p := next processor in ps19.18. if $(F11 = F1) \land (F22 \neq F2)$ then 20.go to line 6. 19. $F21 := F2 \setminus F22$ 20.Since F1= \emptyset and F2= \emptyset , nothing happens when these lines 21.22.are executed. 23.42 24.25. end

		1.	function first-fit(ts : set of tasks; ps : set of processors)
1.	Form sets $H1$, $H2$, $F1$, $F2$	9	return set of tasks
2.		2. 3.	assigned_tasks := \emptyset If ps consists of type-1 (type-2) processors, then order
	$\forall p: \mathbf{U}[\mathbf{p}] := 0$ $\forall m: \boldsymbol{\tau}[\mathbf{p}] := \emptyset$	0.	ts by decreasing U_i^2/U_i^1 (resp., incr. U_i^1/U_i^2).
	$\forall p: \tau[p] := \emptyset$		Use any order for processors ps, but maintain it
4.	if first-fit($H1$, P^1) $\neq H1$ then declare FAILURE		during the execution of the function first-fit.
5.	if first-fit($H2, P^2$) $\neq H2$ then declare FAILURE	4.	$\tau_i :=$ first task in ts
6.	$F11 := $ first-fit($F1, P^1$)	5.	p := first processor in ps
7.	$F22 := \text{first-fit}(F2, P^2)$	6.	Let k denote the type of processor p (either 1 or 2)
8.	if $(F11 = F1) \land (F22 = F2)$ then declare SUCCESS	7.	if $U[p] + U_i^k \le 1$ then
9.	if $(F11 \neq F1) \land (F22 \neq F2)$ then declare FAILURE	8.	$U[p] := U[p] + U_i^k$
10.	if $(F11 \neq F1) \land (F22 = F2)$ then	9. 10.	$\tau[\mathbf{p}] \coloneqq \tau[\mathbf{p}] \cup \{\tau_i\}$ assigned_tasks := assigned_tasks $\cup \{\tau_i\}$
11.	$F12 := F1 \setminus F11$	11.	if remaining tasks exist in ts then
12.	if first-fit($F12$, P^2) = $F12$ then	12.	$\tau_i := \text{next task in ts}$
13.	declare SUCCESS	13.	go to line 5.
14.	else	14.	else
15.	declare FAILURE	15.	return assigned_tasks
16.	end	16. 17.	end if else
	end	18.	if remaining processors exist in ps then
	if $(F11 = F1) \land (F22 \neq F2)$ then	19.	p := next processors on ps
19.		20.	go to line 6.
20.		01	
20.21.	The eleventithes to		ninataa hara
$\frac{21}{22}$.	The algorithm te	en	ninales nere.
22.			
$\frac{23}{24}$.	end		43
	end		
20.	chu		

Theorem 1: The speed competitive ratio of FF-3C is at most two.

A task set τ is feasible on a computing platform $\pi \rightarrow$ FF-3C schedules τ on the computing platform 2* π

Algorithm FF-4C and FF-4C-NTC and FF-4C-COMB: like FF-3C but with improved average-case performance

- Formulate the problem as <u>Integer Linear Program</u>
 - Minimize U subject to:
 - 1. $\sum_{j=1}^{m} x_{i,j} = 1,$ (i = 1,2,...,n)2. $\sum_{i=1}^{n} (x_{i,j} * u_{i,j}) <= U,$ (j = 1,2,...,m)3. $x_{i,j} = 0 \text{ or } x_{i,j} = 1$ (i = 1,2,...,n); (j = 1,2,...,m)

- Formulate the problem as <u>Integer Linear Program</u>
 - Minimize U subject to:
 - 1. $\sum_{j=1}^{m} x_{i,j} = 1,$ (i = 1, 2, ..., n)2. $\sum_{i=1}^{n} (x_{i,j} * u_{i,j}) <= U,$ (j = 1, 2, ..., m)3. $x_{i,j} = 0 \text{ or } x_{i,j} = 1$ (i = 1, 2, ..., n); (j = 1, 2, ..., m)
 - <u>NP-complete</u>: cannot be solved in polynomial time

- Formulate the problem as <u>Integer Linear Program</u>
 - Minimize U subject to:
 - 1. $\sum_{j=1}^{m} x_{i,j} = 1,$ (i = 1,2,...,n)2. $\sum_{i=1}^{n} (x_{i,j} * u_{i,j}) <= U,$ (j = 1,2,...,m)3. $x_{i,j} = 0 \text{ or } x_{i,j} = 1$ (i = 1,2,...,n); (j = 1,2,...,m)
 - <u>NP-complete</u>: cannot be solved in polynomial time
- Relax it to Linear Programming
 - 3. $0 \le x_{i,j} \le 1$ (i = 1,2,...,n); (j = 1,2,...,m)
 - Solvable in polynomial time
 - <u>At most 'm' fractional tasks</u>

- Formulate the problem as <u>Integer Linear Program</u>
 - Minimize U subject to:
 - 1. $\sum_{j=1}^{m} x_{i,j} = 1,$ (i = 1, 2, ..., n)2. $\sum_{i=1}^{n} (x_{i,j} * u_{i,j}) <= U,$ (j = 1, 2, ..., m)3. $x_{i,j} = 0 \text{ or } x_{i,j} = 1$ (i = 1, 2, ..., n); (j = 1, 2, ..., m)
 - <u>NP-complete</u>: cannot be solved in polynomial time
- Relax it to Linear Programming
 - 3. $0 \le x_{i,j} \le 1$ (i = 1,2,...,n); (j = 1,2,...,m)
 - Solvable in polynomial time
 - <u>At most 'm' fractional tasks</u>
- Assign the fractional tasks integrally
 - Exhaustive enumeration (RTAS04)
 - Bi-partite matching (ICPP04)

Average-case performance evaluation

Comparison of three algorithms (Y-Axis: log₁₀ scale)



Necessary Multiplication Factor

Average-case performance evaluation

		New Al	lgorithms		Old Algorithms							
		Measured a	avg exec tim	e	Meas	sured avg exec ti	me incl CPLEX or	/erhead	Measured avg exec time incl CPLEX overhead - avg CPLEX overhead			
Multiplication	FF-3C	FF-4C	FF-4C	FF-4C	SKB-RTAS	SKB-RTAS	SKB-ICPP	SKB-ICPP	SKB-RTAS	SKB-RTAS	SKB-ICPP	SKB-ICPP
factor			-NTC	-COMB		-IMP		-IMP		-IMP		-IMP
1.00	0.85	0.76	0.93	1.08	32481.61	32545.39	394715.80	369120.15	14324.45	14388.23	164603.39	161727.00
1.25	0.52	0.52	0.51	0.53	31657.49	31572.03	393758.65	325045.97	13500.33	13414.87	163646.24	149405.05
1.50	0.49	0.49	0.45	0.48	31751.65	31729.69	381899.86	297359.20	13594.49	13572.52	161185.38	140149.17
1.75	0.47	0.46	0.42	0.46	31744.69	31582.66	337182.98	290084.67	13587.52	13425.49	151049.23	137254.26
2.00	0.49	0.48	0.40	0.48	31736.95	31768.30	291714.93	287719.46	13579.79	13611.13	137972.10	136531.41

Table 1. Comparison of average execution time of algorithms (in microseconds)

Conclusions

- + Bin-packing is possible, with good performance, on heterogeneous multiprocessors with two types of processors.
- + Such bin-packing performs well.

Conclusions

- + Bin-packing is possible, with good performance, on heterogeneous multiprocessors with two types of processors.
- + Such bin-packing performs well:
 - * FF-3C has speed competitive ratio at most two;
 - * FF-4C-COMB has speed competitive ratio at most two;
 - * FF-4C-COMB requires on average processors of lower speed than the previously best algorithm;
 - * FF-4C-COMB runs more than 10000 times faster than previously best known algorithm.

Recent extensions to the work

• **Theorem 2**: The speed competitive ratio of FF-3C is at most 1/(1-a)

- 'a' is the maximum utilization of a task

FF-4C and FF-4C-NTC and FF-4C-COMB

 like FF-3C but with improved average-case
 performance

Thank You!