Stakeholder Cooperation for Improved Predictability and Lower Cost Remote Services

Joen Dahlberg, Tatiana Polishchuk, Valentin Polishchuk, **Christiane Schmidt**

LINKÖPING UNIVERSITY

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports
    - Labour accounts for up to 85% of ATS cost

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports
    - Labour accounts for up to 85% of ATS cost

  ➡ Significant cost savings for small airports (30-120movements a day)

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports
    - Labour accounts for up to 85% of ATS cost

  ➡ Significant cost savings for small airports (30-120movements a day)
- To ensure safety: No simultaneous movements at airports controlled from the same module

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports
  - Labour accounts for up to 85% of ATS cost

  ➡ Significant cost savings for small airports (30-120movements a day)
- To ensure safety: No simultaneous movements at airports controlled from the same module

➡ In extreme case in Sweden: simultaneous movements at all five airports

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports
    - Labour accounts for up to 85% of ATS cost
  - ➡ Significant cost savings for small airports (30-120movements a day)
- To ensure safety: No simultaneous movements at airports controlled from the same module

➡ In extreme case in Sweden: simultaneous movements at all five airports

- ➡ Each airport needs separate RTM

- Remotely operated towers enable control of multiple aerodromes from a single Remote Tower Module (RTM) in a Remote Tower Center.
- In Sweden: two remotely controlled airports in operation, five more studied.
- Splits the cost of Air Traffic Services (ATS) provision and staff management between several airports
  - Labour accounts for up to 85% of ATS cost
  - ➡ Significant cost savings for small airports (30-120movements a day)
- To ensure safety: No simultaneous movements at airports controlled from the same module

➡ In extreme case in Sweden: simultaneous movements at all five airports

  ➡ Each airport needs separate RTM

➡ **Possibilities to perturb flight schedules?** (current flight schedules consider only the single airport, ATCO might have to put a/c on hold anyhow…)

Problem Formulation

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

	04:00												05:00												06:00											
	0	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*	*0*	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*	*0*	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*
AP1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

   ❖ Row per airport (a)

	04:00												05:00												06:00											
	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

  - ❖ Row per airport (a)
  - ❖ Column per each slot (s)

	04:00												05:00												06:00											
	0	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*	*0*	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*	*0*	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*
AP1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

  - ❖ Row per airport (a)
  - ❖ Column per each slot (s)
  - ❖ F_{as} = 1 if a movement happens at airport a at time slot s

	04:00											
	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	0	1	1	0	0	0	0	0	1	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0

	05:00											
	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	1	1	1	0	0	0	0	0	1	0	0
AP2	1	1	0	1	0	1	0	0	0	1	0	0
AP3	0	1	1	0	0	0	0	0	0	1	0	1
AP4	0	0	0	0	0	1	0	0	0	0	0	0
AP5	0	1	1	1	0	0	0	0	0	1	0	0

	06:00											
	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	1	0	1	0	0	0	0	0	0
AP4	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	0	1	0	0	1	0	0	0	0	0

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

  ❖ Row per airport (a)
  ❖ Column per each slot (s)
  ❖ F_{as} = 1 if a movement happens at airport a at time slot s
  ❖ F_{as} = 0 otherwise

	04:00												05:00												06:00											
	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

- ❖ Row per airport (a)
- ❖ Column per each slot (s)
- ❖ $F_{as} = 1$ if a movement happens at airport a at time slot s
- ❖ $F_{as} = 0$ otherwise

	04:00												05:00												06:00											
	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0

- ***Conflict:*** two movements during the same slot in different airports (in F: two 1s in the same column)

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

- ❖ Row per airport (a)
- ❖ Column per each slot (s)
- ❖ F_{as} = 1 if a movement happens at airport a at time slot s
- ❖ F_{as} = 0 otherwise

	04:00												05:00												06:00											
	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55	0	5	10	15	20	25	30	35	40	45	50	55
AP1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0

- ***Conflict:*** two movements during the same slot in different airports (in F: two 1s in the same column)
- Conflicting airports should never be assigned to the same RTM

- Input: Aircraft movements at each airport from Demand Data Repository (DDR) hosted by EUROCONTROL
- Split the time into 5-min intervals, called ***slots*** , and put every flight into its slot

➡ Input matrix F:

- ❖ Row per airport (a)
- ❖ Column per each slot (s)
- ❖ F_{as} = 1 if a movement happens at airport a at time slot s
- ❖ F_{as} = 0 otherwise

	04:00											
	0	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*
AP1	0	0	1	1	0	0	0	0	0	1	0	0
AP2	0	0	1	0	1	0	0	1	0	1	1	0
AP3	0	0	0	0	0	1	0	1	0	0	0	0
AP4	0	0	0	1	0	1	0	0	0	0	1	0
AP5	0	0	1	1	0	0	0	1	0	0	1	0

	05:00											
	0	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*
AP1	0	1	1	1	0	0	0	0	0	1	0	0
AP2	1	1	0	1	0	1	0	0	0	1	0	0
AP3	0	1	1	0	0	0	0	0	0	1	0	1
AP4	0	0	0	0	0	1	0	0	0	0	0	0
AP5	0	1	1	1	0	0	0	0	0	1	0	0

	06:00											
	0	*5*	*10*	*15*	*20*	*25*	*30*	*35*	*40*	*45*	*50*	*55*
AP1	0	0	0	1	0	0	1	0	0	0	0	0
AP2	0	0	0	1	1	0	1	0	0	0	0	0
AP3	0	0	0	1	0	1	0	0	0	0	0	0
AP4	1	0	1	1	0	0	0	0	1	1	1	0
AP5	0	0	0	1	0	0	1	0	0	0	0	0

- ***Conflict:*** two movements during the same slot in different airports (in F: two 1s in the same column)
- Conflicting airports should never be assigned to the same RTM

Conflict count	AP1	AP2	AP3	AP4	AP5
AP1		1058	621	366	339
AP2	1058		6473	3400	3021
AP3	621	6473		2603	2517
AP4	366	3400	2603		1449
AP5	339	3021	2517	1449	

Conflict days	AP1	AP2	AP3	AP4	AP5
AP1		341	316	278	285
AP2	341		366	363	365
AP3	316	366		362	362
AP4	278	363	362		359
AP5	285	365	362	359	

- Output: Shifted flights and Airport-to-RTM assignment

- Output: Shifted flights and Airport-to-RTM assignment
- Goal: ”small” shifts to the flight schedules → decreased number of required RTMs

- Output: Shifted flights and Airport-to-RTM assignment
- Goal: ”small” shifts to the flight schedules → decreased number of required RTMs
- Measure for shift?

LINKÖPING UNIVERSITY

- Output: Shifted flights and Airport-to-RTM assignment
- Goal: ”small” shifts to the flight schedules → decreased number of required RTMs
- Measure for shift?
    - ❖ Maximum slot shift Δ (in minutes; multiple of 5, as we shift only by whole slots)

- Output: Shifted flights and Airport-to-RTM assignment
- Goal: ”small” shifts to the flight schedules → decreased number of required RTMs
- Measure for shift?
    - ❖ Maximum slot shift Δ (in minutes; multiple of 5, as we shift only by whole slots)
    - ❖ Number of shifts *S*

- Output: Shifted flights and Airport-to-RTM assignment
- Goal: ”small” shifts to the flight schedules → decreased number of required RTMs
- Measure for shift?
  - ❖ Maximum slot shift Δ (in minutes; multiple of 5, as we shift only by whole slots)
  - ❖ Number of shifts *S*
- MAP = maximum number of airports per module

Formal problem definition:

LINKÖPING UNIVERSITY

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most *S* flights are moved

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most *S* flights are moved
- Each flight is moved by at most Δ

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, S
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most S flights are moved
- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most *S* flights are moved
- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most *S* flights are moved
- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module
- At most M modules are used

Decision problem

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most *S* flights are moved
- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module
- At most M modules are used

Decision problem

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

- At most *S* flights are moved
- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module
- At most M modules are used

For optimisation problem: Move one constraint in objective function

Decision problem

Formal problem definition:

Flights Rescheduling and Airport-to-Module Assignment (FRAMA)

Given:

- Flight slots in a set of airports (the matrix F)
- Maximum allowable shift of a flight
- Maximum total number of allowable shifts, *S*
- Maximum number of airports per RTM, MAP
- Total number of modules, M

Find: New slots for the flights and an assignment of airports to RTMs such that

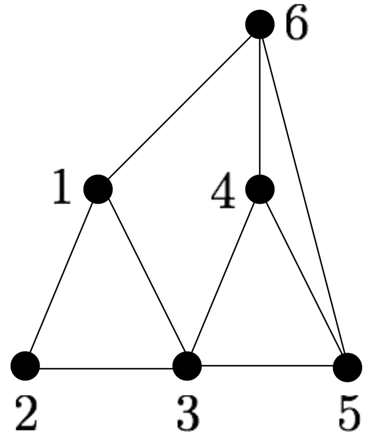
- At most *S* flights are moved
- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module
- At most M modules are used

For optimisation problem: Move one constraint in objective function
For us: Minimize number M of used RTMs, while respecting the bounds Δ, S, MAP

Problem Complexity

Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

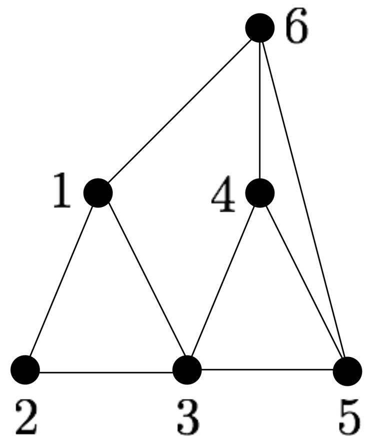
Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.
Proof: Reduction from Partition into Triangles (PIT)



Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

Proof: Reduction from **Partition into Triangles (PIT)**

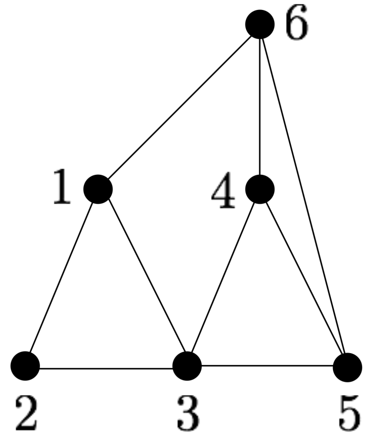
- Graph G = (V,E) (of maximum degree four)



Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

Proof: Reduction from **Partition into Triangles (PIT)**

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

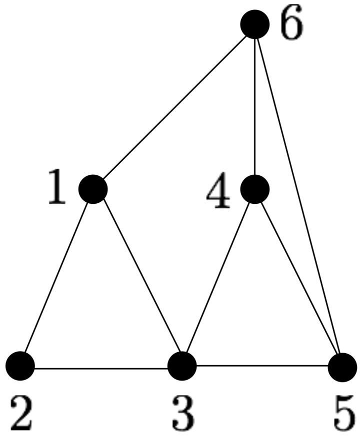


Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:



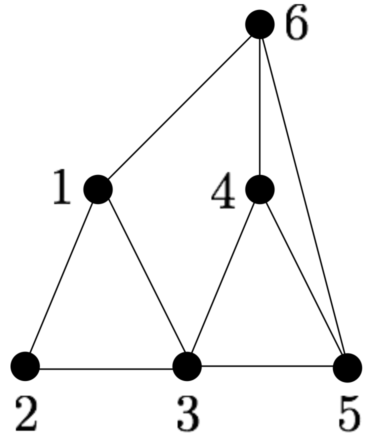
Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

Proof: Reduction from **Partition into Triangles (PIT)**

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows



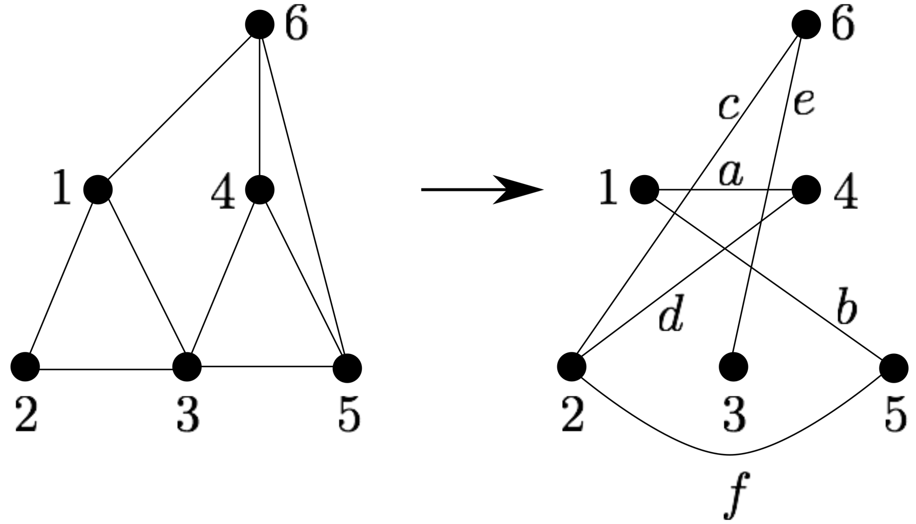
Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1,V_2,\ldots,,V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement)



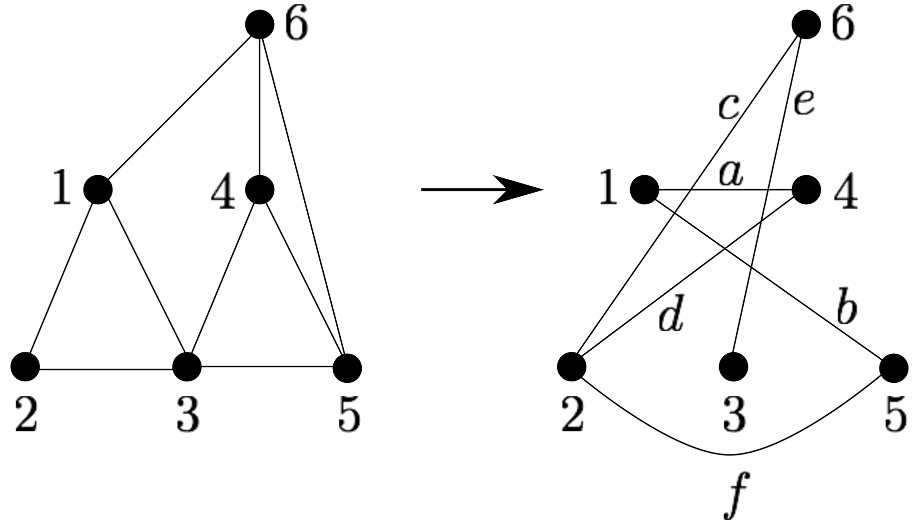
Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

Proof: Reduction from **Partition into Triangles (PIT)**

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V



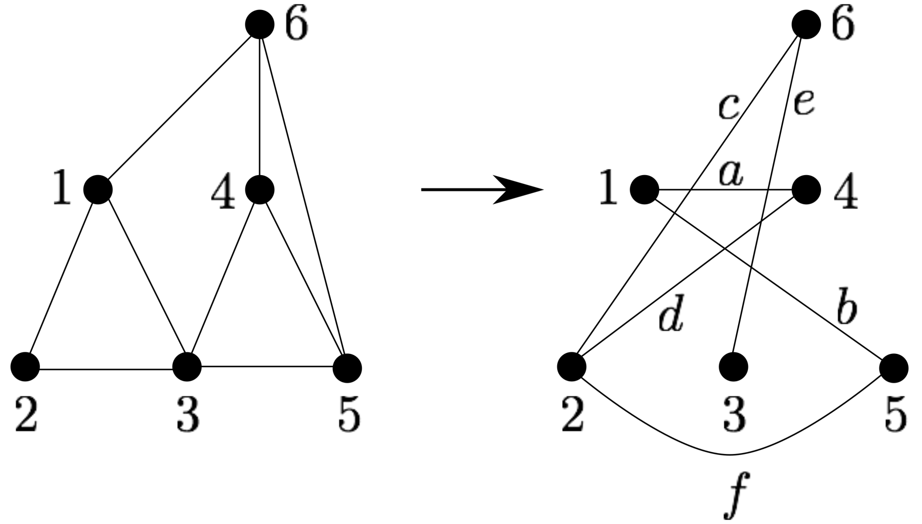
Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V
  → $|E^c \backslash E|$ time slots



	a	b	c	d	e	f
1	1	1	0	0	0	0
2	0	0	1	1	0	1
3	0	0	0	0	1	0
4	1	0	0	1	0	0
5	0	1	0	0	0	0
6	0	0	1	0	1	1

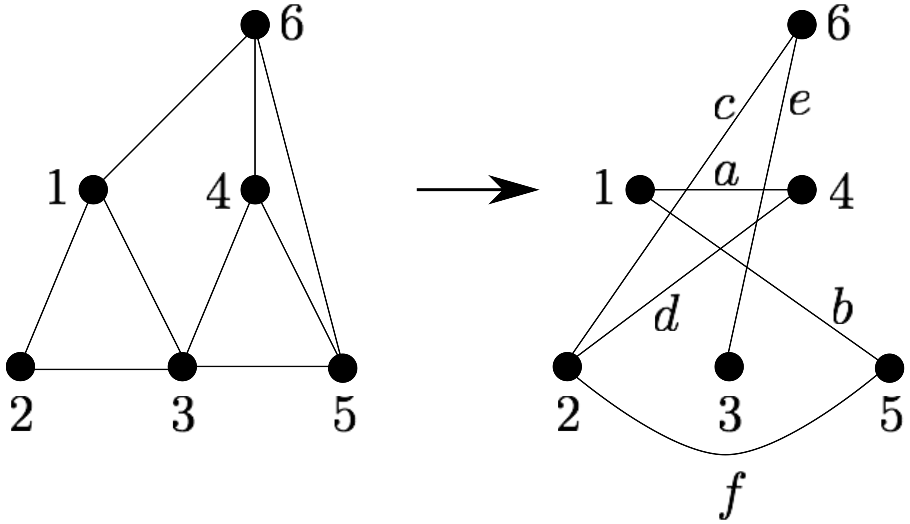
Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V
  → $|E^c \backslash E|$ time slots
- For time slot corresponding to $e^c=\{v,w\} \in E^c \backslash E$ we add two 1's to the time slot column: to the airports of v and w, all other entries in that column are 0's.



	a	b	c	d	e	f
1	1	1	0	0	0	0
2	0	0	1	1	0	1
3	0	0	0	0	1	0
4	1	0	0	1	0	0
5	0	1	0	0	0	0
6	0	0	1	0	1	1

Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

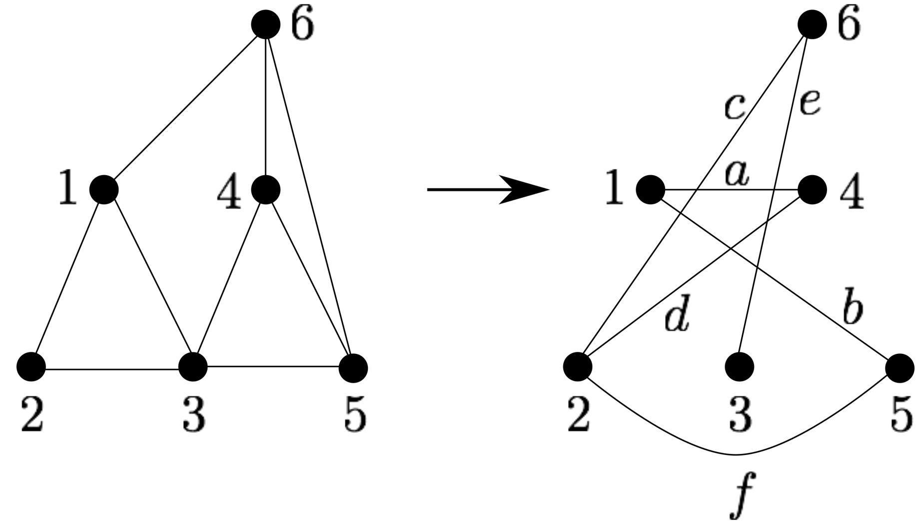
Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c = (V, E^c)$ complete graph on V → $|E^c \setminus E|$ time slots
- For time slot corresponding to $e^c = \{v,w\} \in E^c \setminus E$ we add two 1's to the time slot column: to the airports of v and w, all other entries in that column are 0's.

Any solution to FRAMA with $\Delta = 0$ and MAP=3 groups the airports (vertices) into triples, such that there are no conflicts between any of the three airports in a triple, that is, such that there is an edge between any of the three vertices in the triple.



	a	b	c	d	e	f
1	1	1	0	0	0	0
2	0	0	1	1	0	1
3	0	0	0	0	1	0
4	1	0	0	1	0	0
5	0	1	0	0	0	0
6	0	0	1	0	1	1

LINKÖPING UNIVERSITY

Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

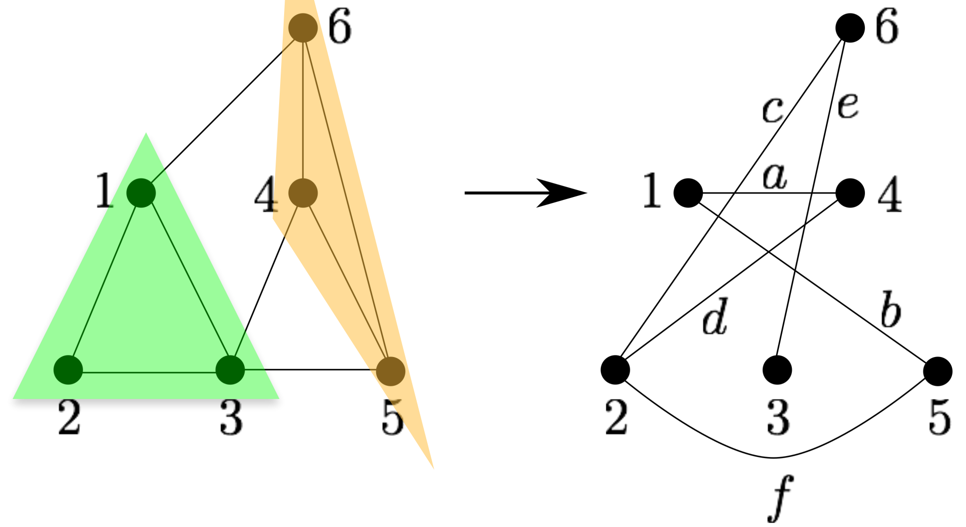
Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V → $|E^c \setminus E|$ time slots
- For time slot corresponding to $e^c=\{v,w\} \in E^c \setminus E$ we add two 1's to the time slot column: to the airports of v and w, all other entries in that column are 0's.

Any solution to FRAMA with $\Delta=0$ and MAP=3 groups the airports (vertices) into triples, such that there are no conflicts between any of the three airports in a triple, that is, such that there is an edge between any of the three vertices in the triple.



	a	b	c	d	e	f
1	1	1	0	0	0	0
2	0	0	1	1	0	1
3	0	0	0	0	1	0
4	1	0	0	1	0	0
5	0	1	0	0	0	0
6	0	0	1	0	1	1

Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

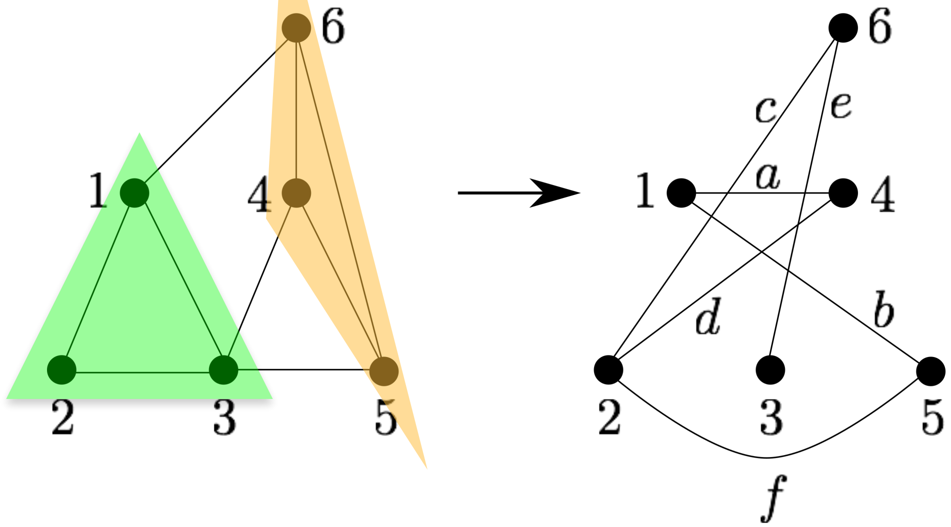
Proof: Reduction from Partition into Triangles (PIT)

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V → $|E^c \setminus E|$ time slots
- For time slot corresponding to $e^c=\{v,w\} \in E^c \setminus E$ we add two 1's to the time slot column: to the airports of v and w, all other entries in that column are 0's.

Any solution to FRAMA with Δ=0 and MAP=3 groups the airports (vertices) into triples, such that there are no conflicts between any of the three airports in a triple, that is, such that there is an edge between any of the three vertices in the triple.



	a	b	c	d	e	f
1	1	1	0	0	0	0
2	0	0	1	1	0	1
3	0	0	0	0	1	0
4	1	0	0	1	0	0
5	0	1	0	0	0	0
6	0	0	1	0	1	1

Theorem: FRAMA is NP-complete, even if $\Delta = 0$ and MAP=3.

Proof: Reduction from **Partition into Triangles (PIT)**

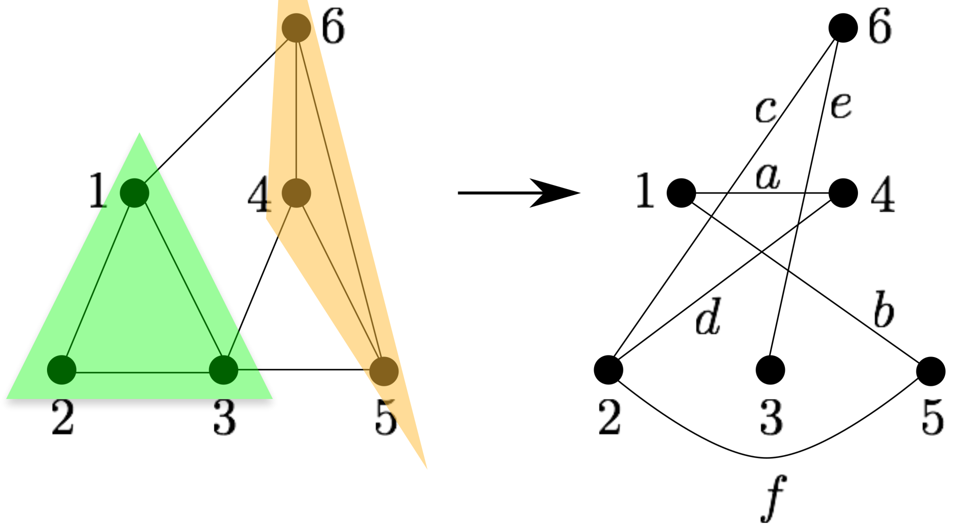
- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V → $|E^c \backslash E|$ time slots
- For time slot corresponding to $e^c=\{v,w\} \in E^c \backslash E$ we add two 1's to the time slot column: to the airports of v and w, all other entries in that column are 0's.

Any solution to FRAMA with $\Delta=0$ and MAP=3 groups the airports (vertices) into triples, such that there are no conflicts between any of the three airports in a triple, that is, such that there is an edge between any of the three vertices in the triple.

➡ We would obtain a solution to PIT



	a	b	c	d	e	f
1	1	1	0	0	0	0
2	0	0	1	1	0	1
3	0	0	0	0	1	0
4	1	0	0	1	0	0
5	0	1	0	0	0	0
6	0	0	1	0	1	1

Theorem: FRAMA is NP-complete, even if Δ= 0 and MAP=3.

Proof: Reduction from **Partition into Triangles (PIT)**

- Graph G = (V,E) (of maximum degree four)
- Can V be partitioned into triples $V_1, V_2, \ldots, V_{|V|/3}$, such that each V_i forms a triangle in G (for each triple of vertices V_i each vertex in V_i is connected to both other vertices in V_i)?

Given an instance of PIT (graph G = (V,E) with max degree four) we construct the matrix F, the input of FRAMA:

- One airport per vertex → F has |V| rows
- Time slot per non-existing edge in G (that is per edge in the G's complement) $G^c=(V,E^c)$ complete graph on V → $|E^c \setminus E|$ time slots
- For time slot corresponding to $e^c=\{v,w\} \in E^c \setminus E$ we add two 1's to the time slot column: to the airports of v and w, all other entries in that column are 0's.

Any solution to FRAMA with Δ=0 and MAP=3 groups the airports (vertices) into triples, such that there are no conflicts between any of the three airports in a triple, that is, such that there is an edge between any of the three vertices in the triple.

➡ We would obtain a solution to PIT

Solution to FRAMA with Δ=0 (and, thus, *S*= 0) and MAP= 3 can be verified in polytime.

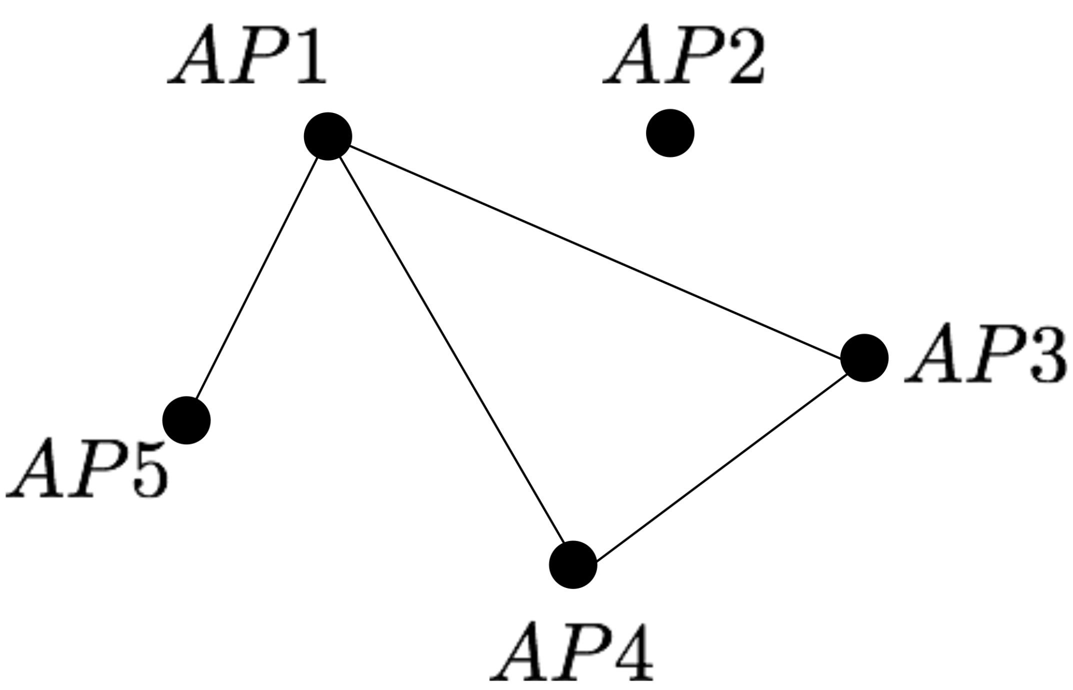
Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

Maximum matching can be found in polynomial time.

Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

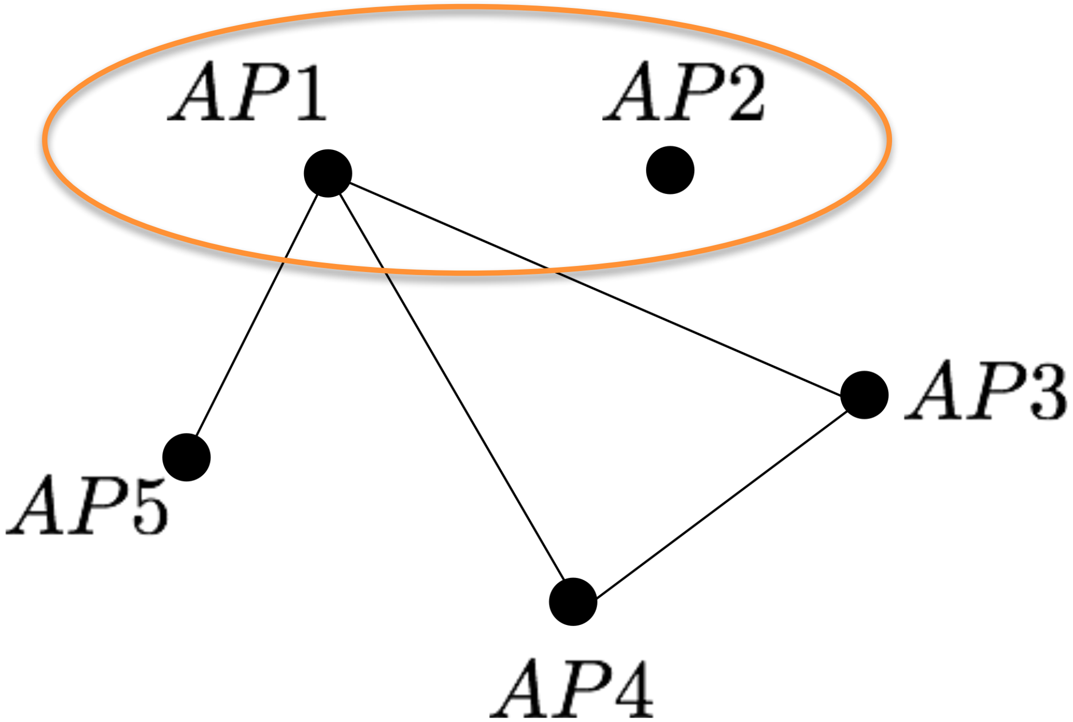
Maximum matching can be found in polynomial time.



Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

Maximum matching can be found in polynomial time.

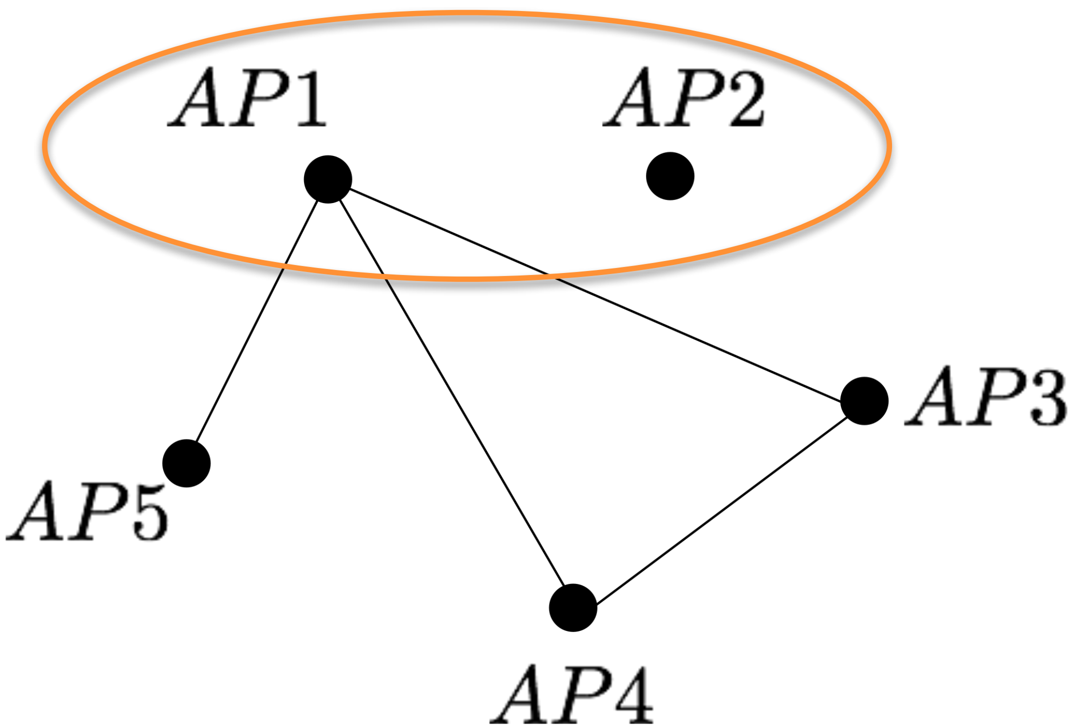
No edge, as AP1 and AP2 are in conflict

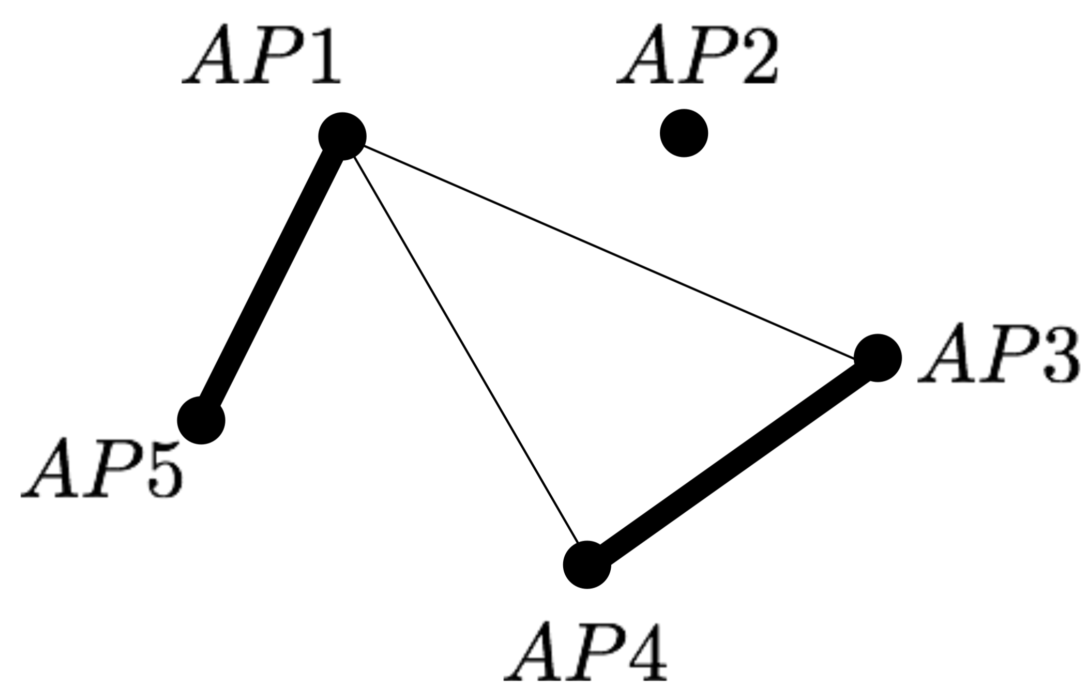


Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

Maximum matching can be found in polynomial time.

No edge, as AP1 and AP2 are in conflict

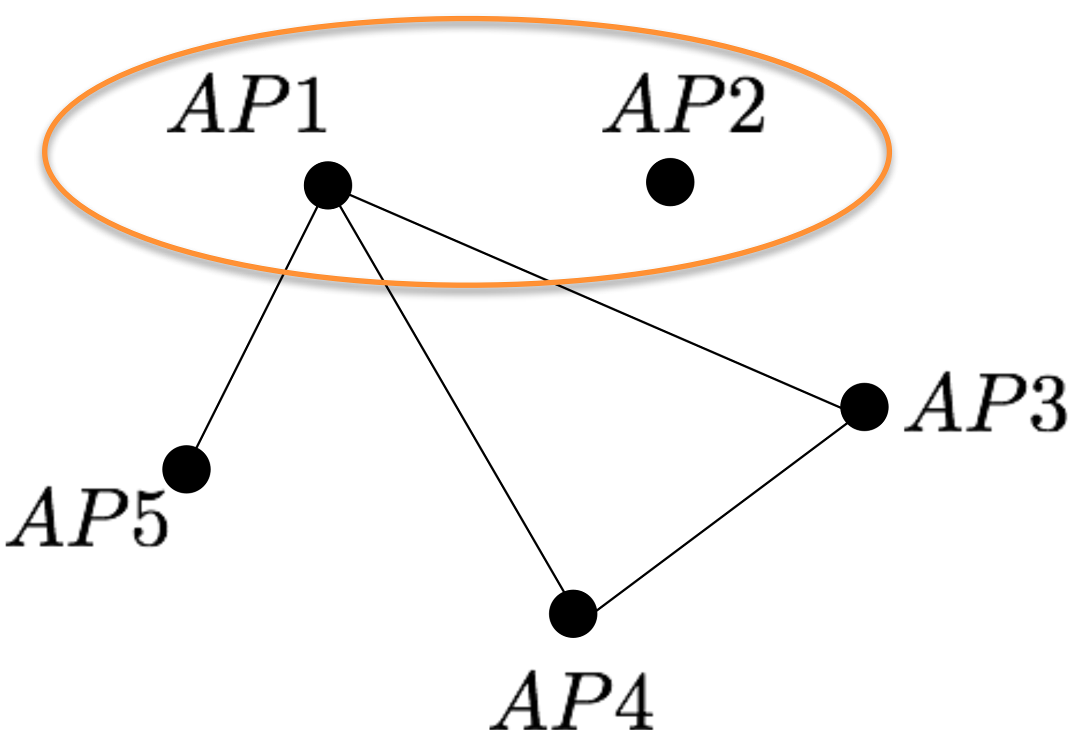


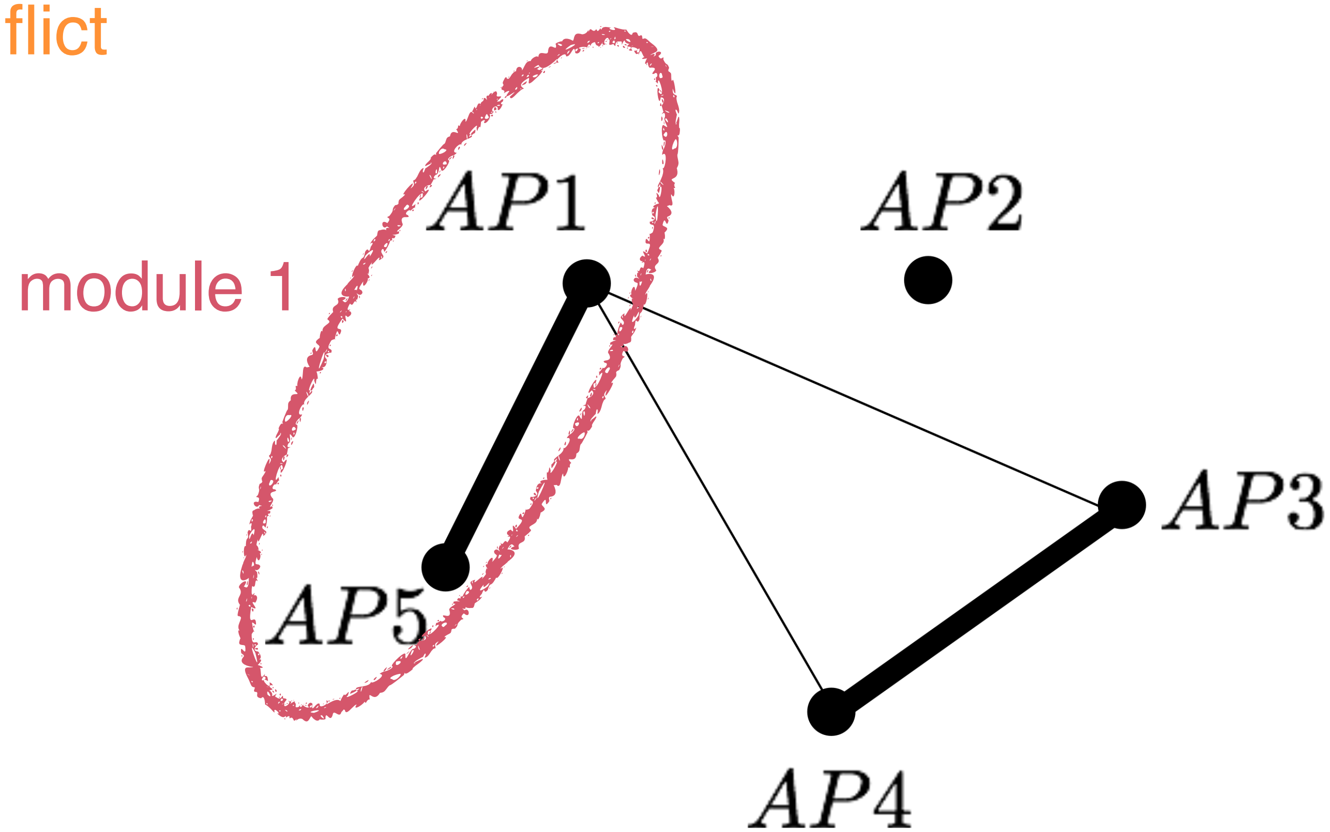


Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

Maximum matching can be found in polynomial time.

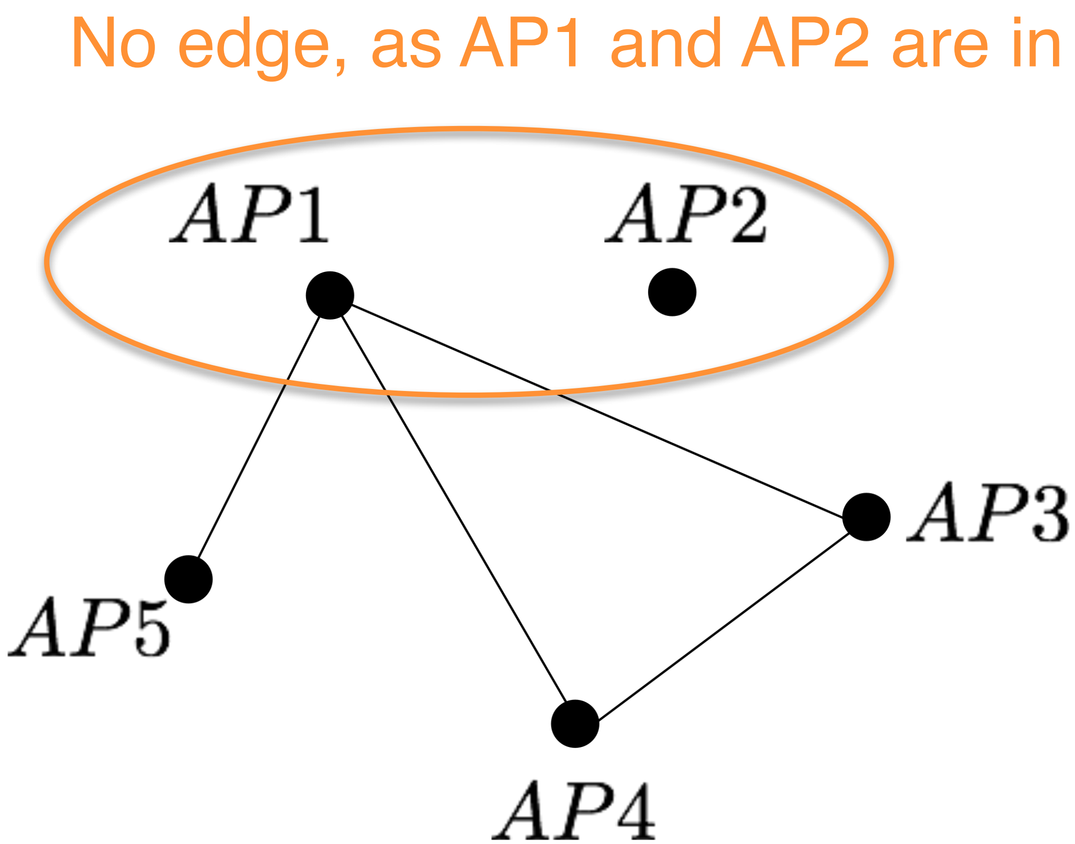
No edge, as AP1 and AP2 are in conflict

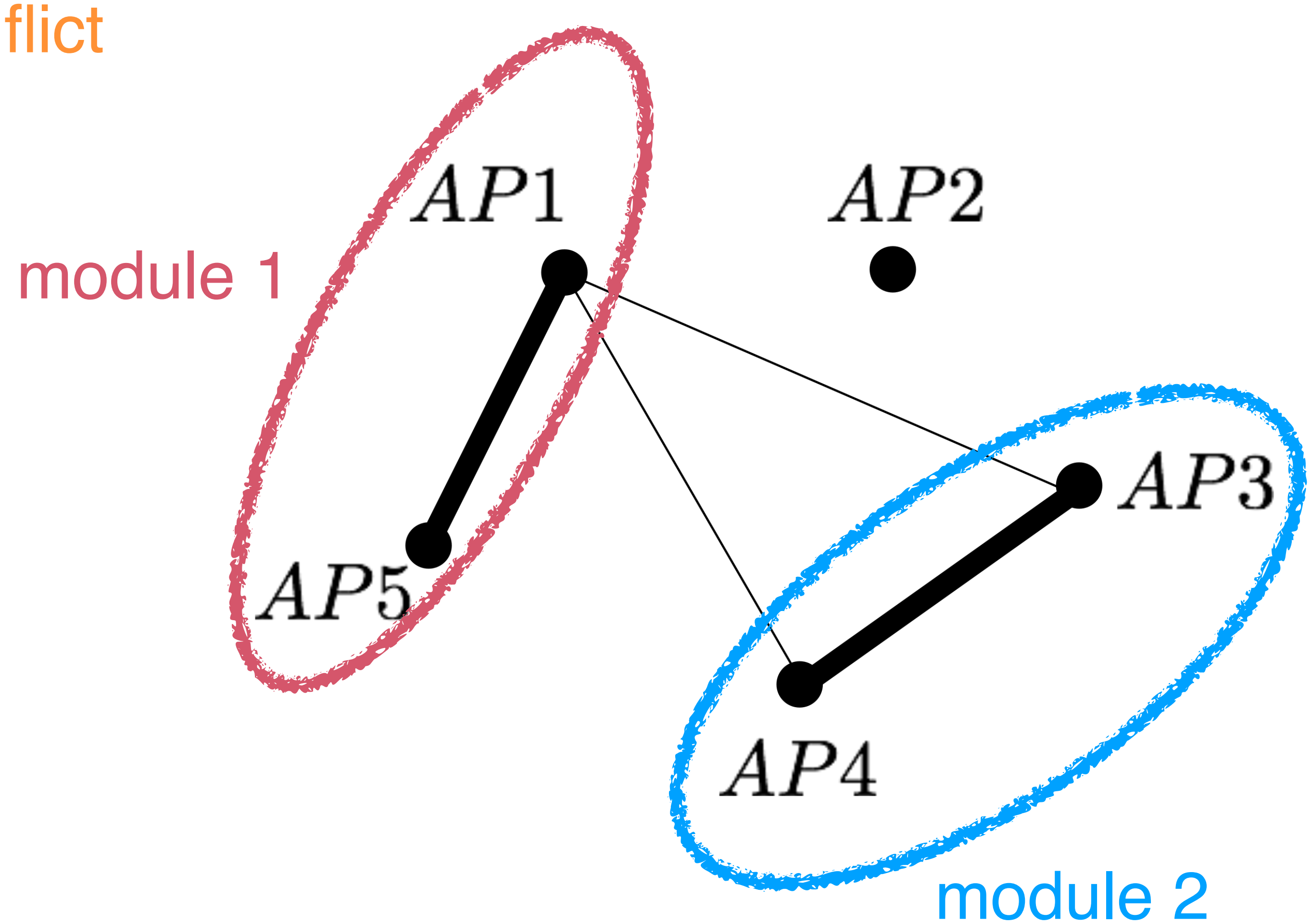




Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

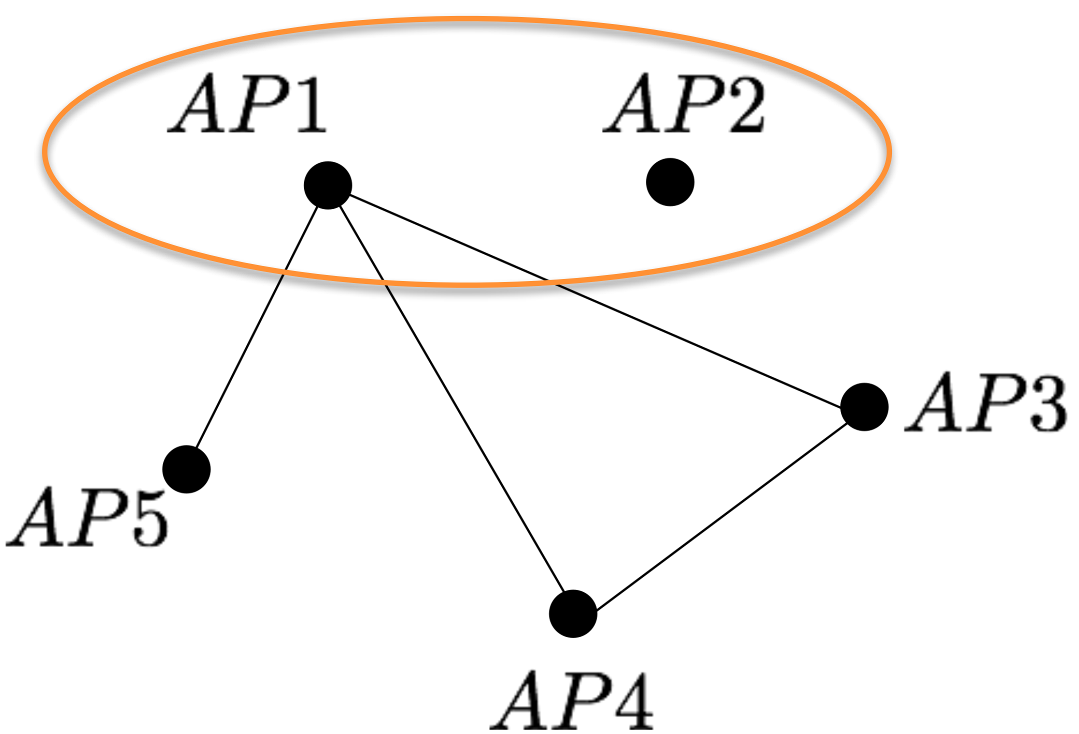
Maximum matching can be found in polynomial time.

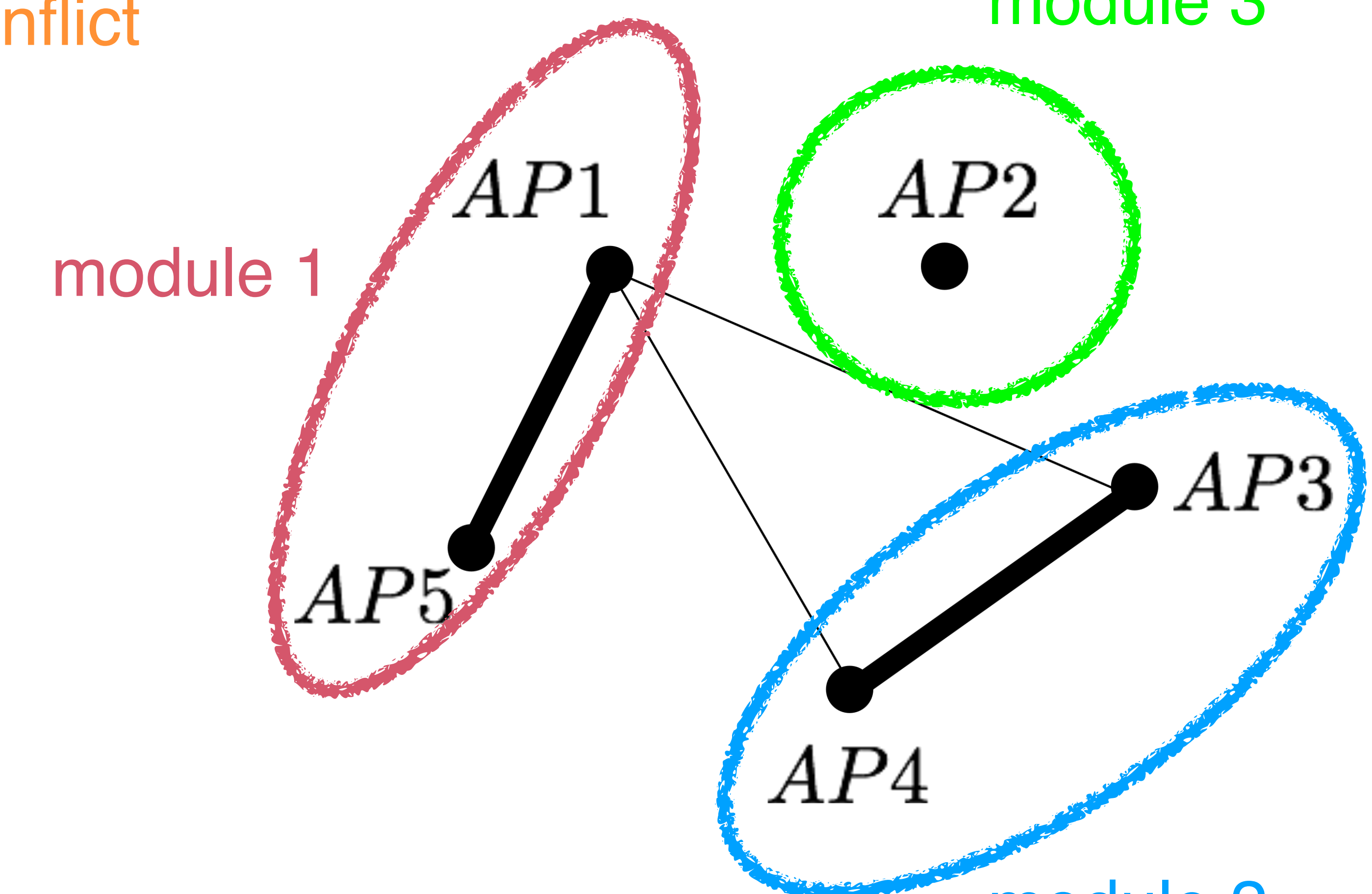




Theorem: For Δ= 0 and MAP=2 Minimizing the number of modules is equivalent to finding a maximum matching in the *airport conflict graph* (vertex for every airport and an edge between two airports if they can be put into the same module).

Maximum matching can be found in polynomial time.





Complexity for Δ> 0 and MAP=2 unknown.

Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

Complexity for $\Delta > 0$ and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts

Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

Complexity for $\Delta > 0$ and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

Complexity for $\Delta > 0$ and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part

Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

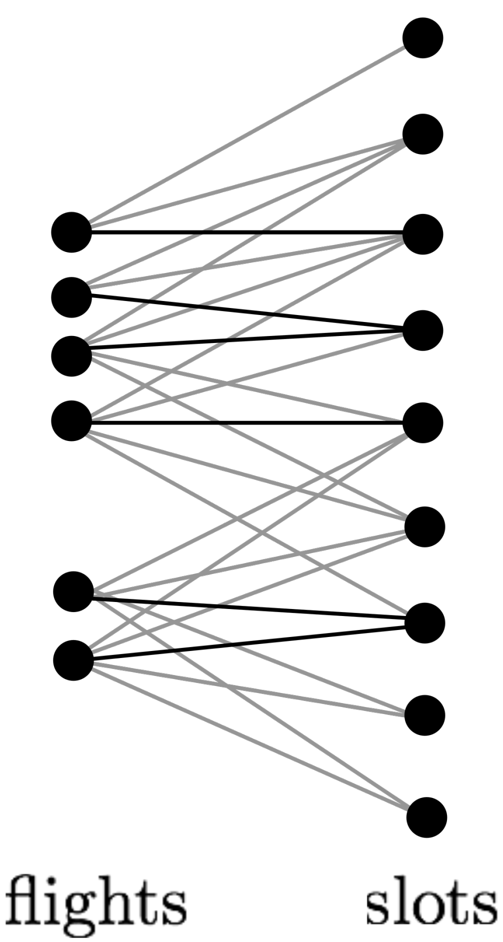
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

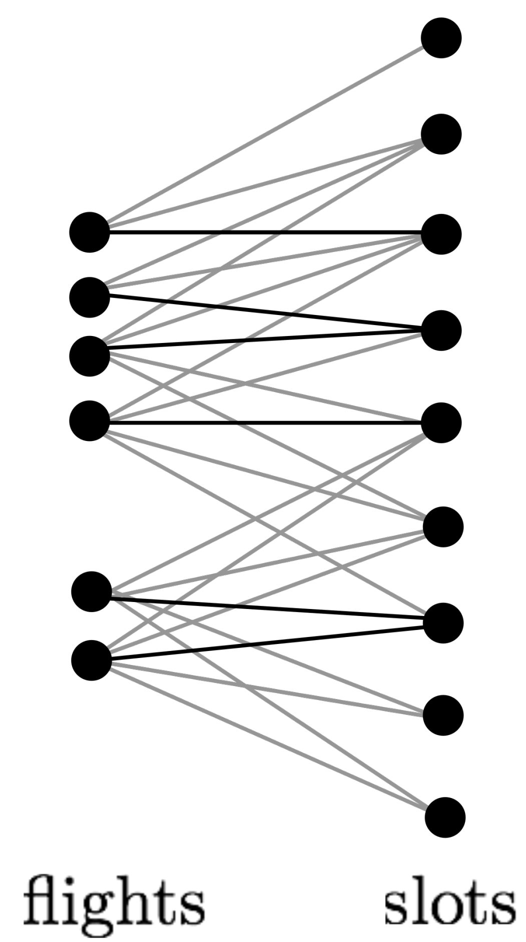
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

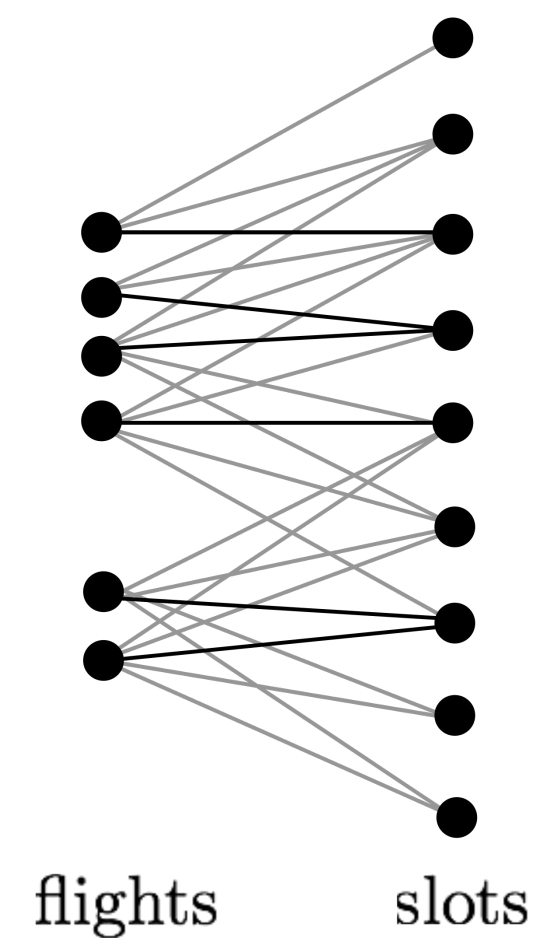
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

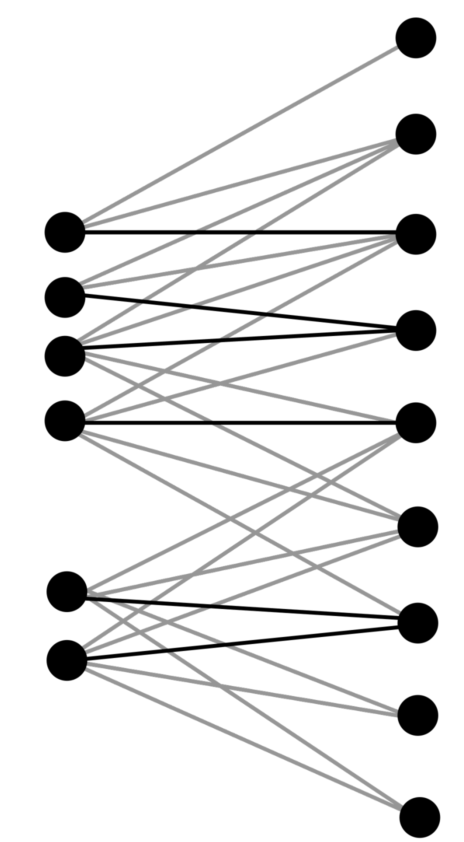
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
    - 0, for edge between flight f and its original slot (black edges)
    - 1, otherwise (gray edges)



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

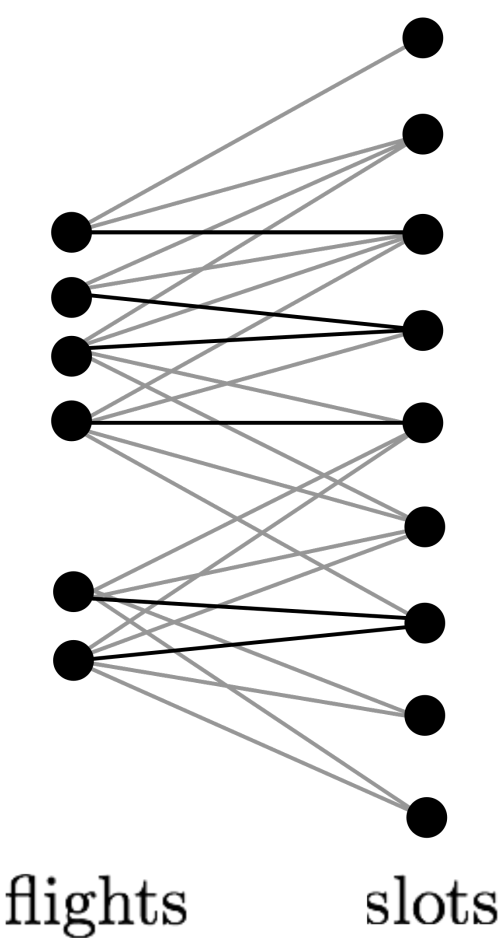
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

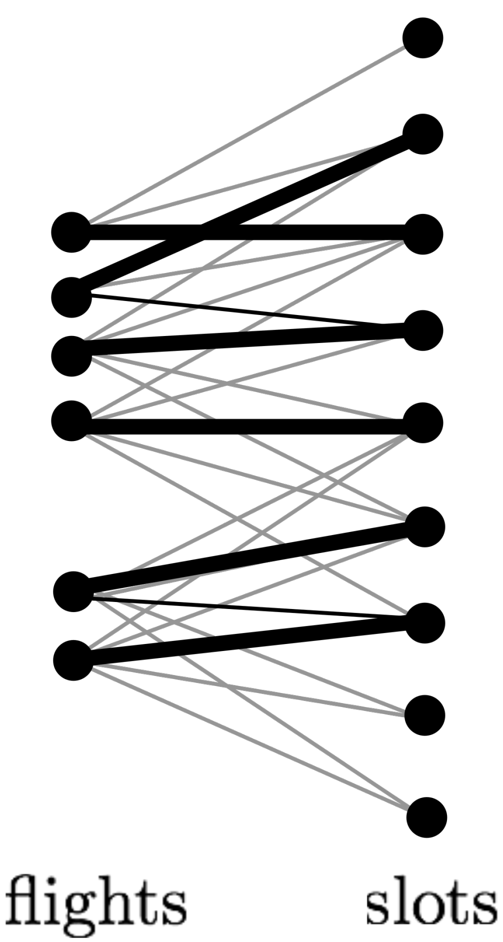
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights
- If no such matching exists, Δ must be increased



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

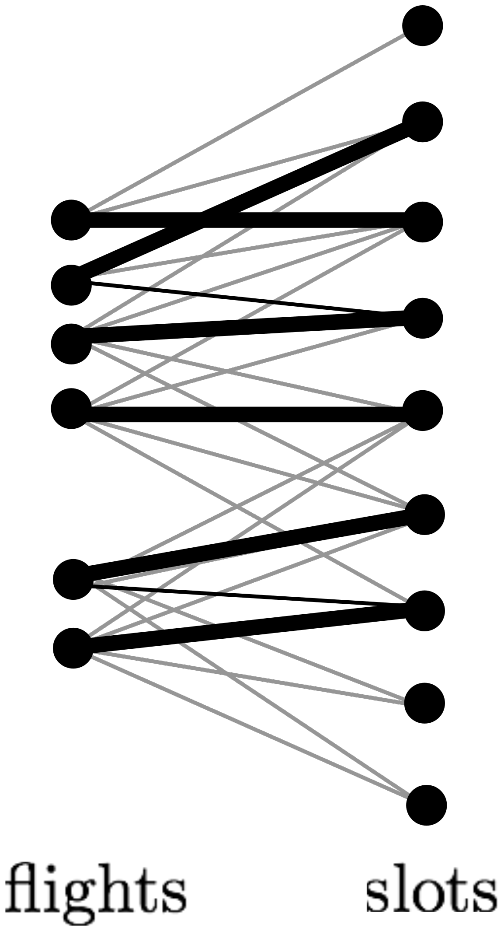
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights
- If no such matching exists, Δ must be increased
- We can also minimize the total amount of shifted minutes: set the weight of each edge equal to the length of the shift



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

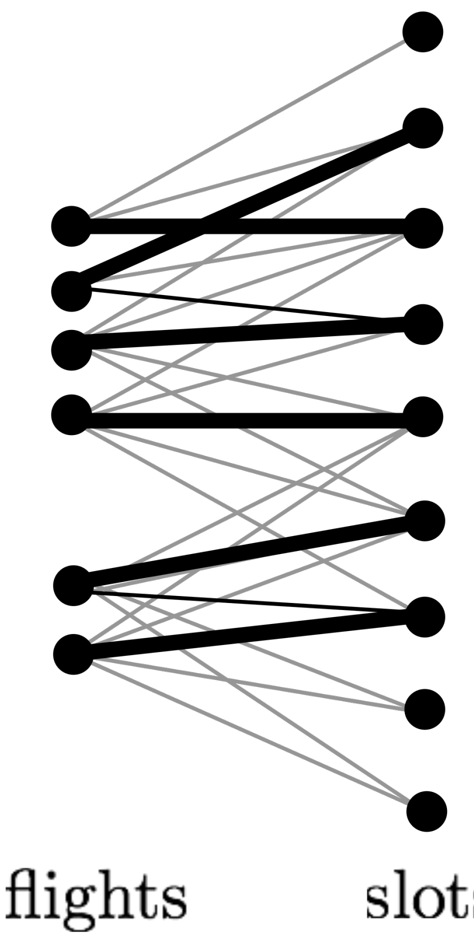
Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights
- If no such matching exists, Δ must be increased
- We can also minimize the total amount of shifted minutes: set the weight of each edge equal to the length of the shift

Runs in polynomial time, but may find suboptimal solutions to FRAMA (not necessary to remove all the conflicts)



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

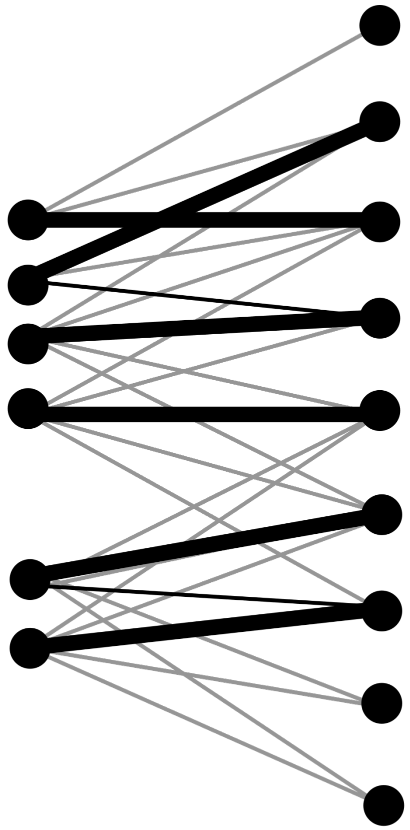
➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights
- If no such matching exists, Δ must be increased
- We can also minimize the total amount of shifted minutes: set the weight of each edge equal to the length of the shift

Runs in polynomial time, but may find suboptimal solutions to FRAMA (not necessary to remove all the conflicts)

For a small number of airports: enumerate all pairs of airports



Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

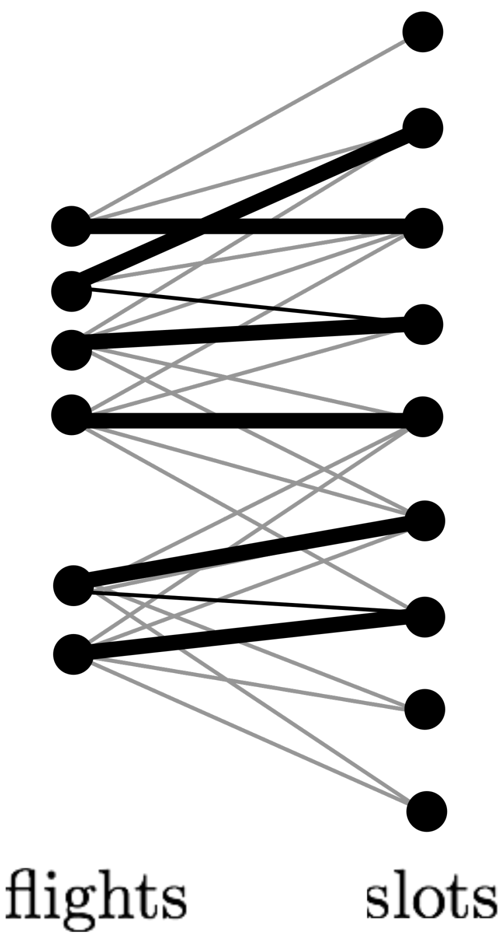
➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights
- If no such matching exists, Δ must be increased
- We can also minimize the total amount of shifted minutes: set the weight of each edge equal to the length of the shift



Runs in polynomial time, but may find suboptimal solutions to FRAMA (not necessary to remove all the conflicts)

For a small number of airports: enumerate all pairs of airports completely eliminate all conflicts for the given pairs (matching) with a given Δ > 0

Complexity for Δ> 0 and MAP=2 unknown.

Possible heuristic:

- First remove all conflicts
- Then assign airports to RTMs

➡ Solve rescheduling and assignment problem separately

Assignment problem is trivial in the absence of conflicts (the airports are arbitrarily packed into the RTMs, with MAP airports per module)

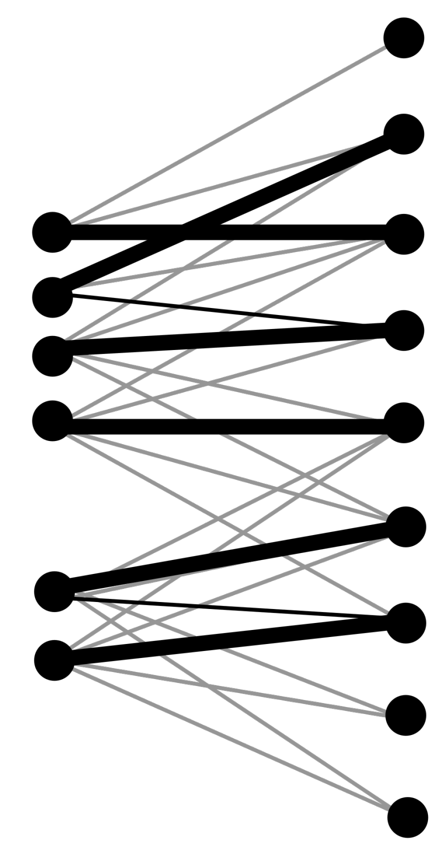
➡ How to deconflict flight schedule?

We can reduce deconfliction problem to matching:

- Bipartite graph: all flights in one part and all slots in the other part
- Flight f is connected to all slots within distance Δ/5 from its original slot
- Edge weight:
  - 0, for edge between flight f and its original slot (black edges)
  - 1, otherwise (gray edges)
- Find the minimum-weight matching in the graph that matches all flights
- If no such matching exists, Δ must be increased
- We can also minimize the total amount of shifted minutes: set the weight of each edge equal to the length of the shift

Runs in polynomial time, but may find suboptimal solutions to FRAMA (not necessary to remove all the conflicts)

For a small number of airports: enumerate all pairs of airports completely eliminate all conflicts for the given pairs (matching) with a given $\Delta > 0$ chose combination with minimum possible number of modules



IP for FRAMA

Decision variables
x_{am}: airport *a* assigned to module *m*
z_m: module *m* is used
y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*
w_{ab}: conflict between airport *a* and airport *b* (some *t*)

A = set of airports
M = set of modules
T = set of time slots
V_a= flights at airport *a*
p_{atf} = cost to move flight *f* at airport *a* to time slot *t*
s_{af} = scheduled time for flight *f* at airport a *i*
δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a= flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a = flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \tag{1}$$

$$
\begin{aligned}
s.t.\ & x_{am} && \leqslant z_m && \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} && = 1 && \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} && \leqslant 1 && \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} && = 1 && \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} && \leqslant 1 + w_{ab} && \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} && \leqslant 2 - w_{ab} && \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} && \leqslant \text{MAP} && \forall m \in M && (8)\\
& x, y, w, z && \textit{binary} && && (9)
\end{aligned}
$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a = flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

c_1*#modules + c_2* sum of shifts

$$
\begin{aligned}
s.t.\ & x_{am} && \leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} && = 1 \quad \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} && \leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} && = 1 \quad \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} && \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} && \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} && \leqslant \text{MAP} \quad \forall m \in M && (8)\\
& x, y, w, z && \quad binary && (9)
\end{aligned}
$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a = flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

c_1*#modules + c_2* sum of shifts

Some airport assigned to module *m*

$$\text{s.t. } x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \quad (2)$$

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \quad (4)$$

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \quad (5)$$

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \quad (6)$$

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \quad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \quad (8)$$

$$x, y, w, z \quad \textit{binary} \quad (9)$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a = flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

c_1*#modules + c_2* sum of shifts

$$s.t. \quad x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \quad (2)$$

Some airport assigned to module *m* →module *m* used

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \quad (4)$$

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \quad (5)$$

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \quad (6)$$

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \quad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \quad (8)$$

$$x, y, w, z \quad \textit{binary} \quad (9)$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a= flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta/5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$
\begin{aligned}
s.t.\quad & x_{am} \leqslant z_m && \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} = 1 && \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} \leqslant 1 && \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 && \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} && \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} \leqslant 2 - w_{ab} && \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} \leqslant \text{MAP} && \forall m \in M && (8)\\
& x, y, w, z \quad \textit{binary} && && (9)
\end{aligned}
$$

(2) Some airport assigned to module *m* →module *m* used

(3) Each airport assigned to 1 module

Decision variables
x_{am}: airport *a* assigned to module *m*
z_m: module *m* is used
y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*
w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$
min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports
M = set of modules
T = set of time slots
V_a= flights at airport *a*
p_{atf} = cost to move flight *f* at airport *a* to time slot *t*
s_{af} = scheduled time for flight *f* at airport a *i*
δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta/5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$s.t.\ x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \quad (2)$$

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \quad (4)$$

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \quad (5)$$

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \quad (6)$$

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \quad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \quad (8)$$

$$x, y, w, z \quad \textit{binary} \quad (9)$$

(2) Some airport assigned to module *m* →module *m* used
(3) Each airport assigned to 1 module
(4) At most 1 flight arrives/departs at airport time slot t

Decision variables
x_{am}: airport *a* assigned to module *m*
z_m: module *m* is used
y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*
w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$
min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports
M = set of modules
T = set of time slots
V_a = flights at airport *a*
p_{atf} = cost to move flight *f* at airport *a* to time slot *t*
s_{af} = scheduled time for flight *f* at airport a *i*
δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta/5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$s.t. \quad x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \quad (2)$$

(2) Some airport assigned to module *m* →module *m* used

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

(3) Each airport assigned to 1 module

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \quad (4)$$

(4) At most 1 flight arrives/departs at airport time slot t

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \quad (5)$$

(5) Each flight ±δ from scheduled time

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \quad (6)$$

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \quad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \quad (8)$$

$$x, y, w, z \quad binary \quad (9)$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a = flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta/5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$s.t. \quad x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \quad (2)$$

(2) Some airport assigned to module *m* →module *m* used

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

(3) Each airport assigned to 1 module

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \quad (4)$$

(4) At most 1 flight arrives/departs at airport time slot t

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \quad (5)$$

(5) Each flight $\pm\delta$ from scheduled time

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \quad (6)$$

(6) Two a/c at same slot at airports *a* and *b*

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \quad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \quad (8)$$

$$x, y, w, z \quad \textit{binary} \quad (9)$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

A = set of airports

M = set of modules

T = set of time slots

V_a= flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$\text{s.t. } x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \quad (2)$$

(2) Some airport assigned to module *m* → module *m* used

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

(3) Each airport assigned to 1 module

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \quad (4)$$

(4) At most 1 flight arrives/departs at airport time slot t

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \quad (5)$$

(5) Each flight $\pm\delta$ from scheduled time

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \quad (6)$$

(6) Two a/c at same slot at airports *a* and *b* → two airports in conflict

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \quad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \quad (8)$$

$$x, y, w, z \quad \textit{binary} \quad (9)$$

Decision variables

x_{am}: airport *a* assigned to module *m*

z_m: module *m* is used

y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*

w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$

min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports

M = set of modules

T = set of time slots

V_a= flights at airport *a*

p_{atf} = cost to move flight *f* at airport *a* to time slot *t*

s_{af} = scheduled time for flight *f* at airport a *i*

δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

$$\min c_1 \sum_{m\in M} z_m + c_2 \sum_{a\in A}\sum_{t\in T}\sum_{f\in V_a} p_{atf}y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$\text{s.t. } x_{am} \leqslant z_m \quad \forall(a,m)\in A\times M \quad (2)$$

(2) Some airport assigned to module *m* → module *m* used

$$\sum_{m\in M} x_{am} = 1 \quad \forall a \in A \quad (3)$$

(3) Each airport assigned to 1 module

$$\sum_{f\in V_a} y_{atf} \leqslant 1 \quad \forall(a,t)\in A\times T \quad (4)$$

(4) At most 1 flight arrives/departs at airport time slot t

$$\sum_{t=\max(1,s_{af}-\delta)}^{\min(|T|,s_{af}+\delta)} y_{atf} = 1 \quad \forall(a,f)\in A\times V_a \quad (5)$$

(5) Each flight ±δ from scheduled time

$$\sum_{f\in V_a} y_{atf} + \sum_{f\in V_b} y_{btf} \leqslant 1 + w_{ab} \forall(a,b,t)\in A\times A\times T, a<b \quad (6)$$

(6) Two a/c at same slot at airports *a* and *b* → two airports in conflict

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall(a,b,m)\in A\times A\times M, a<b \quad (7)$$

(7) If ∃ conflict → airports not same module

$$\sum_{a\in A} x_{am} \leqslant \text{MAP} \quad \forall m\in M \quad (8)$$

$$x,y,w,z \quad \textit{binary} \quad (9)$$

Decision variables
x_{am}: airport *a* assigned to module *m*
z_m: module *m* is used
y_{atf}: flight *f* arrives/departs at/from airport *a* in time slot *t*
w_{ab}: conflict between airport *a* and airport *b* (some *t*)

min # shifts: $p_{atf}=1$ if $t \neq s_{af}$; $p_{atf}=0$ if $t=s_{af}$
min total amount of shifts: $p_{atf}=|t-s_{af}|$

A = set of airports
M = set of modules
T = set of time slots
V_a = flights at airport *a*
p_{atf} = cost to move flight *f* at airport *a* to time slot *t*
s_{af} = scheduled time for flight *f* at airport a *i*
δ maximum shift distance for scheduled aircraft in terms of time slots: $\delta = \Delta / 5$.

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

(1) c_1*#modules + c_2* sum of shifts

$$\begin{aligned}
s.t.\quad & x_{am} \leqslant z_m && \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} = 1 && \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} \leqslant 1 && \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 && \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} && \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} \leqslant 2 - w_{ab} && \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} \leqslant \text{MAP} && \forall m \in M && (8)\\
& x, y, w, z \quad \textit{binary} && && (9)
\end{aligned}$$

(2) Some airport assigned to module *m* → module *m* used
(3) Each airport assigned to 1 module
(4) At most 1 flight arrives/departs at airport time slot t
(5) Each flight ±δ from scheduled time
(6) Two a/c at same slot at airports *a* and *b* → two airports in conflict
(7) If ∃ conflict → airports not same module
(8) Max MAP airports to each module

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \qquad (1)$$

$$s.t. \quad x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \qquad (2)$$

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \qquad (3)$$

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \qquad (4)$$

$$\sum_{t=\max(1,s_{af}-\delta)}^{\min(|T|,s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \qquad (5)$$

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \qquad (6)$$

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \qquad (7)$$

$$\sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M \qquad (8)$$

$$x, y, w, z \quad \textit{binary} \qquad (9)$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \tag{1}$$

$$
\begin{aligned}
s.t.\ & x_{am} && \leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} && = 1 \quad \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} && \leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} && = 1 \quad \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} && \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} && \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} && \leqslant \mathrm{MAP} \quad \forall m \in M && (8)\\
& x, y, w, z && \quad binary && (9)
\end{aligned}
$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)
We choose c_1 and c_2 such that minimizing the modules is the primary objective: $c_1 >> c_2$

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \tag{1}$$

$$s.t.\ x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M \tag{2}$$

$$\sum_{m \in M} x_{am} = 1 \quad \forall a \in A \tag{3}$$

$$\sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T \tag{4}$$

$$\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a \tag{5}$$

$$\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b \tag{6}$$

$$x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b \tag{7}$$

$$\sum_{a \in A} x_{am} \leqslant \mathrm{MAP} \quad \forall m \in M \tag{8}$$

$$x, y, w, z \quad binary \tag{9}$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)
We choose c_1 and c_2 such that minimizing the modules is the primary objective: $c_1 >> c_2$
IP computes new slots for flights and assigns airports to RTMs, such that:

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

$$
\begin{aligned}
s.t.\ x_{am} &\leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
\sum_{m \in M} x_{am} &= 1 \quad \forall a \in A && (3)\\
\sum_{f \in V_a} y_{atf} &\leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
\sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} &= 1 \quad \forall (a,f) \in A \times V_a && (5)\\
\sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} &\leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
x_{am} + x_{bm} &\leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
\sum_{a \in A} x_{am} &\leqslant \text{MAP} \quad \forall m \in M && (8)\\
x, y, w, z & \quad binary && (9)
\end{aligned}
$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)
We choose c_1 and c_2 such that minimizing the modules is the primary objective: $c_1 >> c_2$
IP computes new slots for flights and assigns airports to RTMs, such that:

- Each flight is moved by at most Δ

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \tag{1}$$

$$
\begin{aligned}
s.t.\ & x_{am} && \leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} && = 1 \quad \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} && \leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} && = 1 \quad \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} && \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} && \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} && \leqslant \text{MAP} \quad \forall m \in M && (8)\\
& x, y, w, z && \quad binary && (9)
\end{aligned}
$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)
We choose c_1 and c_2 such that minimizing the modules is the primary objective: $c_1 >> c_2$
IP computes new slots for flights and assigns airports to RTMs, such that:

- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

$$\begin{aligned}
s.t.\ & x_{am} && \leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} && = 1 \quad \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} && \leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} && = 1 \quad \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} && \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} && \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} && \leqslant \text{MAP} \quad \forall m \in M && (8)\\
& x, y, w, z && \quad binary && (9)
\end{aligned}$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)
We choose c_1 and c_2 such that minimizing the modules is the primary objective: $c_1 >> c_2$
IP computes new slots for flights and assigns airports to RTMs, such that:

- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \quad (1)$$

$$
\begin{aligned}
s.t.\quad & x_{am} && \leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} && = 1 \quad \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} && \leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} && = 1 \quad \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} && \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} && \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} && \leqslant \text{MAP} \quad \forall m \in M && (8)\\
& x, y, w, z && \quad binary && (9)
\end{aligned}
$$

IP formulation of FRAMA optimises c_1*M + c_2*S (could move one in constraint)

We choose c_1 and c_2 such that minimizing the modules is the primary objective: $c_1 >> c_2$

IP computes new slots for flights and assigns airports to RTMs, such that:

- Each flight is moved by at most Δ
- No conflicting airports are assigned to the same RTM
- At most MAP airports are assigned per module

➡ IP formulation solves FRAMA!

$$\min c_1 \sum_{m \in M} z_m + c_2 \sum_{a \in A} \sum_{t \in T} \sum_{f \in V_a} p_{atf} y_{atf} \tag{1}$$

$$\begin{aligned}
s.t.\ & x_{am} \leqslant z_m \quad \forall (a,m) \in A \times M && (2)\\
& \sum_{m \in M} x_{am} = 1 \quad \forall a \in A && (3)\\
& \sum_{f \in V_a} y_{atf} \leqslant 1 \quad \forall (a,t) \in A \times T && (4)\\
& \sum_{t=\max(1, s_{af}-\delta)}^{\min(|T|, s_{af}+\delta)} y_{atf} = 1 \quad \forall (a,f) \in A \times V_a && (5)\\
& \sum_{f \in V_a} y_{atf} + \sum_{f \in V_b} y_{btf} \leqslant 1 + w_{ab} \forall (a,b,t) \in A \times A \times T, a < b && (6)\\
& x_{am} + x_{bm} \leqslant 2 - w_{ab} \forall (a,b,m) \in A \times A \times M, a < b && (7)\\
& \sum_{a \in A} x_{am} \leqslant \text{MAP} \quad \forall m \in M && (8)\\
& x, y, w, z \quad \mathit{binary} && (9)
\end{aligned}$$

Experimental Study

Additional airports considered for remote operation in Sweden:

Additional airports considered for remote operation in Sweden:

- **Airport 1 (AP1):** Small airport with low traffic, few scheduled flights per hour, non-regular helicopter traffic, sometimes special testing activities.
- **Airport 2 (AP2):** Low to medium-sized airport, multiple scheduled flights per hour, regular special traffic flights (usually open 24/7, with exceptions).
- **Airport 3 (AP3):** Small regional airport with regular scheduled flights (usually open 24/7, with exceptions)
- **Airport 4 (AP4):** Small airport with significant seasonal variations.
- **Airport 5 (AP5):** Small airport with low scheduled traffic, non-regular helicopter flights.

Additional airports considered for remote operation in Sweden:

- **Airport 1 (AP1):** Small airport with low traffic, few scheduled flights per hour, non-regular helicopter traffic, sometimes special testing activities.
- **Airport 2 (AP2):** Low to medium-sized airport, multiple scheduled flights per hour, regular special traffic flights (usually open 24/7, with exceptions).
- **Airport 3 (AP3):** Small regional airport with regular scheduled flights (usually open 24/7, with exceptions)
- **Airport 4 (AP4):** Small airport with significant seasonal variations.
- **Airport 5 (AP5):** Small airport with low scheduled traffic, non-regular helicopter flights.

We use traffic data from October 19, 2016—the day with highest traffic in 2016

Additional airports considered for remote operation in Sweden:

- **Airport 1 (AP1):** Small airport with low traffic, few scheduled flights per hour, non-regular helicopter traffic, sometimes special testing activities.
- **Airport 2 (AP2):** Low to medium-sized airport, multiple scheduled flights per hour, regular special traffic flights (usually open 24/7, with exceptions).
- **Airport 3 (AP3):** Small regional airport with regular scheduled flights (usually open 24/7, with exceptions)
- **Airport 4 (AP4):** Small airport with significant seasonal variations.
- **Airport 5 (AP5):** Small airport with low scheduled traffic, non-regular helicopter flights.

We use traffic data from October 19, 2016—the day with highest traffic in 2016
286 flight movements were scheduled on this day for the five airports

Additional airports considered for remote operation in Sweden:

- **Airport 1 (AP1):** Small airport with low traffic, few scheduled flights per hour, non-regular helicopter traffic, sometimes special testing activities.
- **Airport 2 (AP2):** Low to medium-sized airport, multiple scheduled flights per hour, regular special traffic flights (usually open 24/7, with exceptions).
- **Airport 3 (AP3):** Small regional airport with regular scheduled flights (usually open 24/7, with exceptions)
- **Airport 4 (AP4):** Small airport with significant seasonal variations.
- **Airport 5 (AP5):** Small airport with low scheduled traffic, non-regular helicopter flights.

We use traffic data from October 19, 2016—the day with highest traffic in 2016
286 flight movements were scheduled on this day for the five airports
For first set of experiments: without self-conflicts → 233 movements

Additional airports considered for remote operation in Sweden:

- **Airport 1 (AP1):** Small airport with low traffic, few scheduled flights per hour, non-regular helicopter traffic, sometimes special testing activities.
- **Airport 2 (AP2):** Low to medium-sized airport, multiple scheduled flights per hour, regular special traffic flights (usually open 24/7, with exceptions).
- **Airport 3 (AP3):** Small regional airport with regular scheduled flights (usually open 24/7, with exceptions)
- **Airport 4 (AP4):** Small airport with significant seasonal variations.
- **Airport 5 (AP5):** Small airport with low scheduled traffic, non-regular helicopter flights.

We use traffic data from October 19, 2016—the day with highest traffic in 2016
286 flight movements were scheduled on this day for the five airports
For first set of experiments: without self-conflicts → 233 movements
One optimization problem for each pair (Δ, MAP)

Original Traffic

MAP=5

δ	# of modules	# of shifts $= S$	maximum shift (in mins) $= \Delta$
0	5	0	-
1	2	32	5
2	2	27	10
3	2	26	15
4	2	26	-
5	2	26	-
6	2	26	-
7	1	118	35
8	1	108	40
9	1	99	45
10	1	91	50
11	1	85	55
12	1	83	60
13	1	81	65
14	1	79	70
15	1	78	75
16	1	75	80
17	1	75	85
18	1	75	90
19	1	74	95
20	1	74	100
21	1	73	105

We have 12 x 24 = 288 slots for flight movements

➡ with sufficiently large shifts 233 flight movements in single module

Original Traffic

LINKÖPING UNIVERSITY

MAP=5

δ	# of modules	# of shifts $= S$	maximum shift (in mins) $= \Delta$
0	5	0	-
1	2	32	5
2	2	27	10
3	2	26	15
4	2	26	-
5	2	26	-
6	2	26	-
7	1	118	35
8	1	108	40
9	1	99	45
10	1	91	50
11	1	85	55
12	1	83	60
13	1	81	65
14	1	79	70
15	1	78	75
16	1	75	80
17	1	75	85
18	1	75	90
19	1	74	95
20	1	74	100
21	1	73	105

No rescheduling allowed: need 5 RTMs

We have 12 x 24 = 288 slots for flight movements

➡ with sufficiently large shifts 233 flight movements in single module

Original Traffic

LINKÖPING UNIVERSITY

MAP=5

δ	# of modules	# of shifts $= S$	maximum shift (in mins) $= \Delta$
0	5	0	-
1	2	32	5
2	2	27	10
3	2	26	15
4	2	26	-
5	2	26	-
6	2	26	-
7	1	118	35
8	1	108	40
9	1	99	45
10	1	91	50
11	1	85	55
12	1	83	60
13	1	81	65
14	1	79	70
15	1	78	75
16	1	75	80
17	1	75	85
18	1	75	90
19	1	74	95
20	1	74	100
21	1	73	105

No rescheduling allowed: need 5 RTMs
Reschedule at most ±5 minutes: 2 RTMs

We have 12 x 24 = 288 slots for flight movements

➡ with sufficiently large shifts 233 flight movements in single module

Original Traffic

LINKÖPING UNIVERSITY

MAP=5

δ	# of modules	# of shifts $= S$	maximum shift (in mins) $= \Delta$
0	5	0	-
1	2	32	5
2	2	27	10
3	2	26	15
4	2	26	-
5	2	26	-
6	2	26	-
7	1	118	35
8	1	108	40
9	1	99	45
10	1	91	50
11	1	85	55
12	1	83	60
13	1	81	65
14	1	79	70
15	1	78	75
16	1	75	80
17	1	75	85
18	1	75	90
19	1	74	95
20	1	74	100
21	1	73	105

No rescheduling allowed: need 5 RTMs

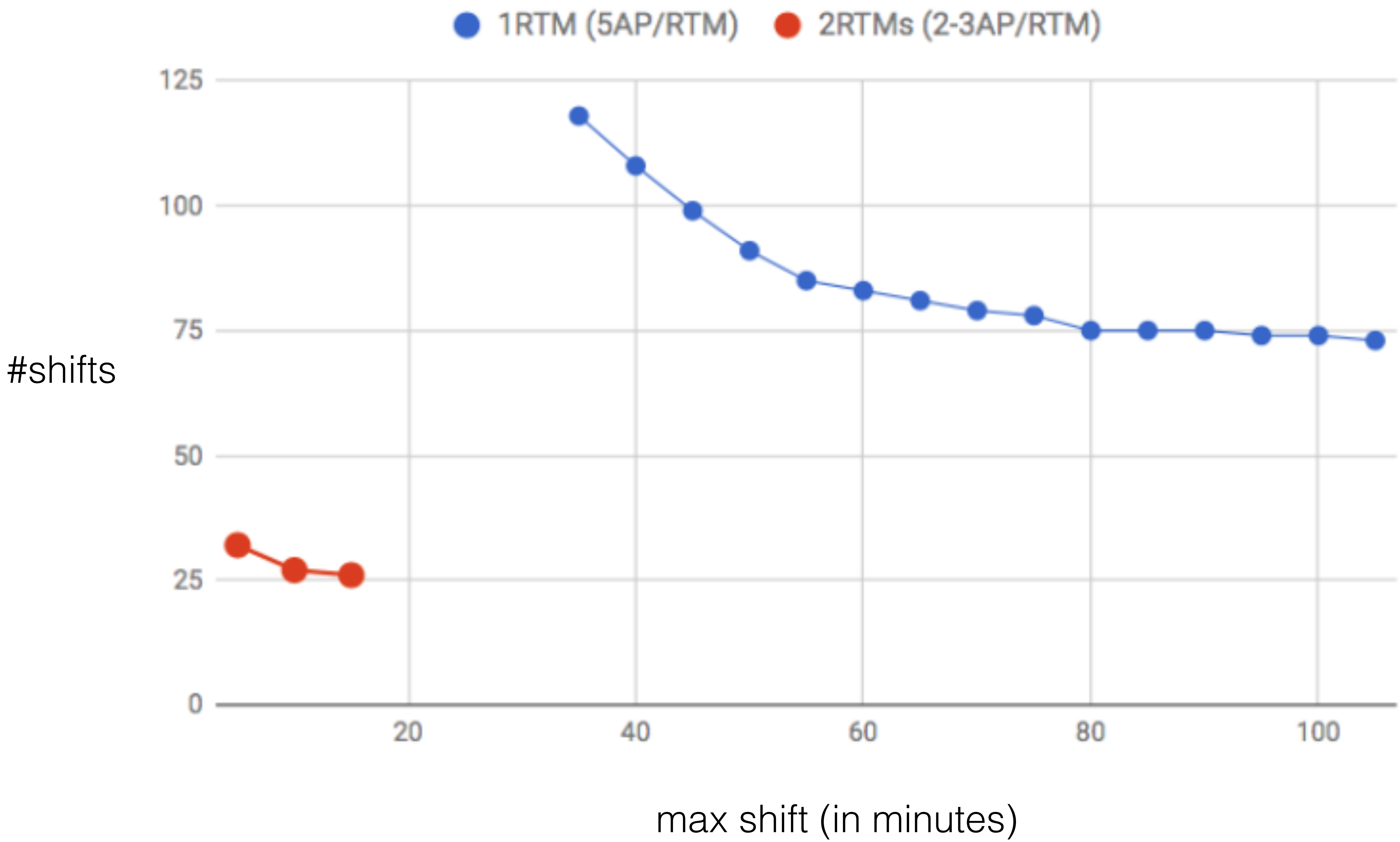
Reschedule at most ±5 minutes: 2 RTMs

For 1 RTM: we need to reschedule by ±35 mins

We have 12 x 24 = 288 slots for flight movements

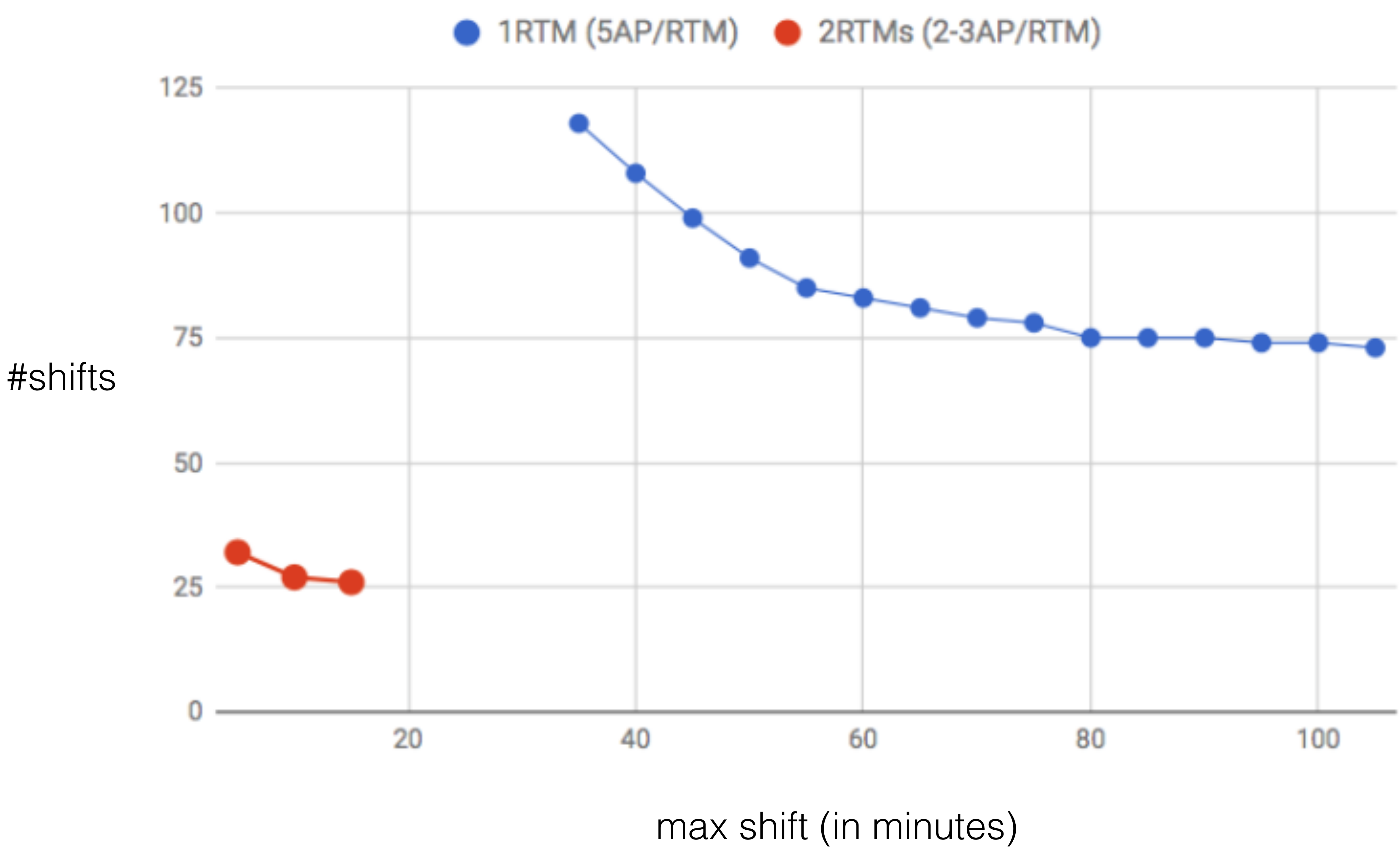
➡ with sufficiently large shifts 233 flight movements in single module

Original Traffic



Original Traffic

Shows tradeoffs: more shifts — larger shifts (more minutes) — more APs/module



Original Traffic

MAP=4

δ	M	S
0	5	0
1	2	32
2	2	27
3	2	26

MAP=3

δ	M	S
0	5	0
1	2	32

MAP=2

δ	# of modules	# of shifts
0	5	0
1	3	7

All 286 movements

In case of a self-induced conflict: model shifts either of them

All 286 movements

In case of a self-induced conflict: model shifts either of them
➡ we start with possible more than one flight movement per time slot and airport

All 286 movements

In case of a self-induced conflict: model shifts either of them

➡ we start with possible more than one flight movement per time slot and airport

➡ δ=0 infeasible by definition

All 286 movements

In case of a self-induced conflict: model shifts either of them
➡ we start with possible more than one flight movement per time slot and airport
➡ δ=0 infeasible by definition

MAP=5

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
36	2	79
37	1	158
38	1	154

All 286 movements

In case of a self-induced conflict: model shifts either of them

➡ we start with possible more than one flight movement per time slot and airport

➡ δ=0 infeasible by definition

MAP=5

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
36	2	79
37	1	158
38	1	154

For 233 movs 2 RTMs were enough for δ=1, now δ=2

All 286 movements

In case of a self-induced conflict: model shifts either of them

➡ we start with possible more than one flight movement per time slot and airport

➡ δ=0 infeasible by definition

MAP=5

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
36	2	79
37	1	158
38	1	154

For 233 movs 2 RTMs were enough for δ=1, now δ=2

For 233 movs 1RTM was enough for δ=7, now δ=37

All 286 movements

LINKÖPING UNIVERSITY

In case of a self-induced conflict: model shifts either of them
➡ we start with possible more than one flight movement per time slot and airport
➡ δ=0 infeasible by definition

MAP=5

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
36	2	79
37	1	158
38	1	154

For 233 movs 2 RTMs were enough for δ=1, now δ=2

For 233 movs 1RTM was enough for δ=7, now δ=37

MAP=4

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
288	2	79

All 286 movements

In case of a self-induced conflict: model shifts either of them

➡ we start with possible more than one flight movement per time slot and airport

➡ δ=0 infeasible by definition

MAP=5

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
36	2	79
37	1	158
38	1	154

For 233 movs 2 RTMs were enough for δ=1, now δ=2

For 233 movs 1RTM was enough for δ=7, now δ=37

MAP=4

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
288	2	79

MAP=3

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
288	2	79

All 286 movements

In case of a self-induced conflict: model shifts either of them
➡ we start with possible more than one flight movement per time slot and airport
➡ δ=0 infeasible by definition

MAP=5

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
36	2	79
37	1	158
38	1	154

For 233 movs 2 RTMs were enough for δ=1, now δ=2

For 233 movs 1RTM was enough for δ=7, now δ=37

MAP=4

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
288	2	79

MAP=3

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	2	103
3	2	80
4	2	79
288	2	79

MAP=2

δ	M	S
0	infeasible	infeasible
1	infeasible	infeasible
2	3	61
3	3	61
4	3	60
288	3	60

Computation times: Solve in two steps

Computation times: Solve in two steps

We solve two optimisation with $c_2 = 0$ and $c_1 = 0$ respectively and fix the $\sum z_k$ to the optimal number of modules used when solving the second optimization problem.

Computation times: Solve in two steps

We solve two optimisation with $c_2 = 0$ and $c_1 = 0$ respectively and fix the $\sum z_k$ to the optimal number of modules used when solving the second optimization problem.

MAP=5

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,40
3	2	80	1,26
4	2	79	1,79
36	2	79	7,97
37	1	158	8,42
38	1	154	9,34
39	1	151	40,84
40	1	149	46,61
41	1	147	45,12
42	1	144	38,10
43	1	141	40,20
44	1	139	43,57
45	1	137	9,24
46	1	136	106,31
47	1	135	148,79
48	1	134	100,03
49	1	133	94,08
50	1	132	479,12
51	1	130	433,79
52	1	128	348,83
53	1	126	11,65
288	1	126	46,49

Computation times: Solve in two steps

We solve two optimisation with $c_2 = 0$ and $c_1 = 0$ respectively and fix the $\sum z_k$ to the optimal number of modules used when solving the second optimization problem.

MAP=5

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,40
3	2	80	1,26
4	2	79	1,79
36	2	79	7,97
37	1	158	8,42
38	1	154	9,34
39	1	151	40,84
40	1	149	46,61
41	1	147	45,12
42	1	144	38,10
43	1	141	40,20
44	1	139	43,57
45	1	137	9,24
46	1	136	106,31
47	1	135	148,79
48	1	134	100,03
49	1	133	94,08
50	1	132	479,12
51	1	130	433,79
52	1	128	348,83
53	1	126	11,65
288	1	126	46,49

MAP=4

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,31
3	2	80	1,06
4	2	79	1,22
288	2	79	60,92

Computation times: Solve in two steps

We solve two optimisation with $c_2 = 0$ and $c_1 = 0$ respectively and fix the $\sum z_k$ to the optimal number of modules used when solving the second optimization problem.

MAP=5

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,40
3	2	80	1,26
4	2	79	1,79
36	2	79	7,97
37	1	158	8,42
38	1	154	9,34
39	1	151	40,84
40	1	149	46,61
41	1	147	45,12
42	1	144	38,10
43	1	141	40,20
44	1	139	43,57
45	1	137	9,24
46	1	136	106,31
47	1	135	148,79
48	1	134	100,03
49	1	133	94,08
50	1	132	479,12
51	1	130	433,79
52	1	128	348,83
53	1	126	11,65
288	1	126	46,49

MAP=4

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,31
3	2	80	1,06
4	2	79	1,22
288	2	79	60,92

MAP=3

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,36
3	2	80	1,28
4	2	79	1,09
288	2	79	51,79

Computation times: Solve in two steps

LINKÖPING UNIVERSITY

We solve two optimisation with $c_2 = 0$ and $c_1 = 0$ respectively and fix the $\sum z_k$ to the optimal number of modules used when solving the second optimization problem.

MAP=5

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,40
3	2	80	1,26
4	2	79	1,79
36	2	79	7,97
37	1	158	8,42
38	1	154	9,34
39	1	151	40,84
40	1	149	46,61
41	1	147	45,12
42	1	144	38,10
43	1	141	40,20
44	1	139	43,57
45	1	137	9,24
46	1	136	106,31
47	1	135	148,79
48	1	134	100,03
49	1	133	94,08
50	1	132	479,12
51	1	130	433,79
52	1	128	348,83
53	1	126	11,65
288	1	126	46,49

MAP=4

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,31
3	2	80	1,06
4	2	79	1,22
288	2	79	60,92

MAP=3

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	2	103	1,36
3	2	80	1,28
4	2	79	1,09
288	2	79	51,79

MAP=2

δ	# of modules	# of shifts $= S$	computation time in sec
0	infeasible	-	-
1	infeasible	-	-
2	3	61	0,55
3	3	61	1,09
4	3	60	0,98
288	3	60	100,30

Increased Traffic Volume

Increased Traffic Volume

Duplicate each of the original flight movements

Increased Traffic Volume

Duplicate each of the original flight movements
Shift randomly by plus/minus one hour

Increased Traffic Volume

Duplicate each of the original flight movements
Shift randomly by plus/minus one hour
Shift again, randomly, by plus/minus 15 minutes

Increased Traffic Volume

LINKÖPING UNIVERSITY

Duplicate each of the original flight movements
Shift randomly by plus/minus one hour
Shift again, randomly, by plus/minus 15 minutes
If two flight movements end up in the same slot, one of the movements is deleted

Increased Traffic Volume

Duplicate each of the original flight movements
Shift randomly by plus/minus one hour
Shift again, randomly, by plus/minus 15 minutes
If two flight movements end up in the same slot, one of the movements is deleted
“2x” data created from all data of the year 2016

Increased Traffic Volume

Duplicate each of the original flight movements
Shift randomly by plus/minus one hour
Shift again, randomly, by plus/minus 15 minutes
If two flight movements end up in the same slot, one of the movements is deleted
“2x” data created from all data of the year 2016
➡ shifted duplicates of flights from October 18, 2016 and October 20, 2016 may now happen on October 19, 2016

Increased Traffic Volume

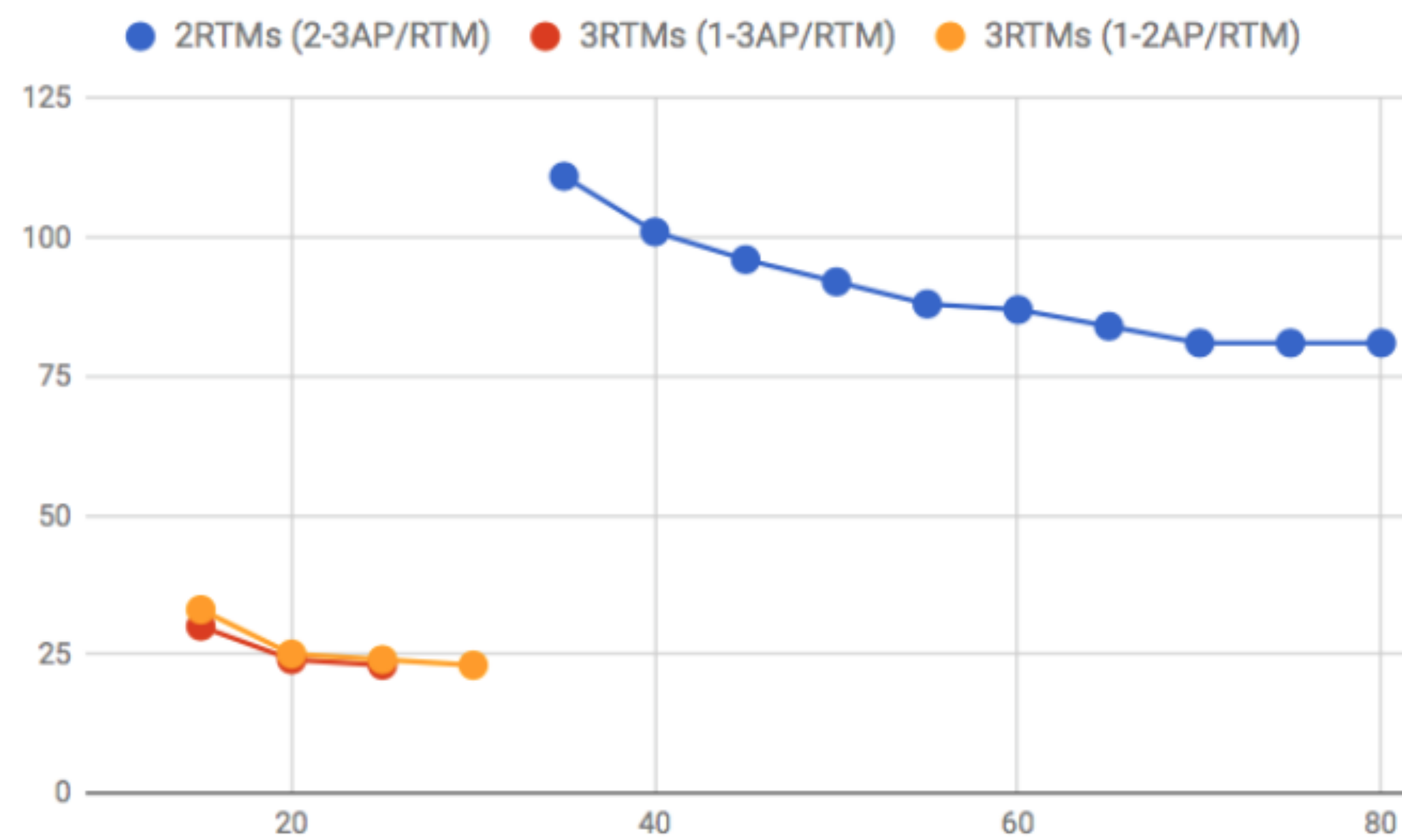
Duplicate each of the original flight movements
Shift randomly by plus/minus one hour
Shift again, randomly, by plus/minus 15 minutes
If two flight movements end up in the same slot, one of the movements is deleted
“2x” data created from all data of the year 2016
➡ shifted duplicates of flights from October 18, 2016 and October 20, 2016 may now happen on October 19, 2016
➡ Not exactly twice the number of movements

Increased Traffic Volume

Duplicate each of the original flight movements
Shift randomly by plus/minus one hour
Shift again, randomly, by plus/minus 15 minutes
If two flight movements end up in the same slot, one of the movements is deleted
“2x” data created from all data of the year 2016
➡ shifted duplicates of flights from October 18, 2016 and October 20, 2016 may now happen on October 19, 2016
➡ Not exactly twice the number of movements
- October 19: data set has 416 flight movements (after deleting double movements in time slots) out of 575 flight movements (all of the movements from 2016 that the duplication and shifting process produces)

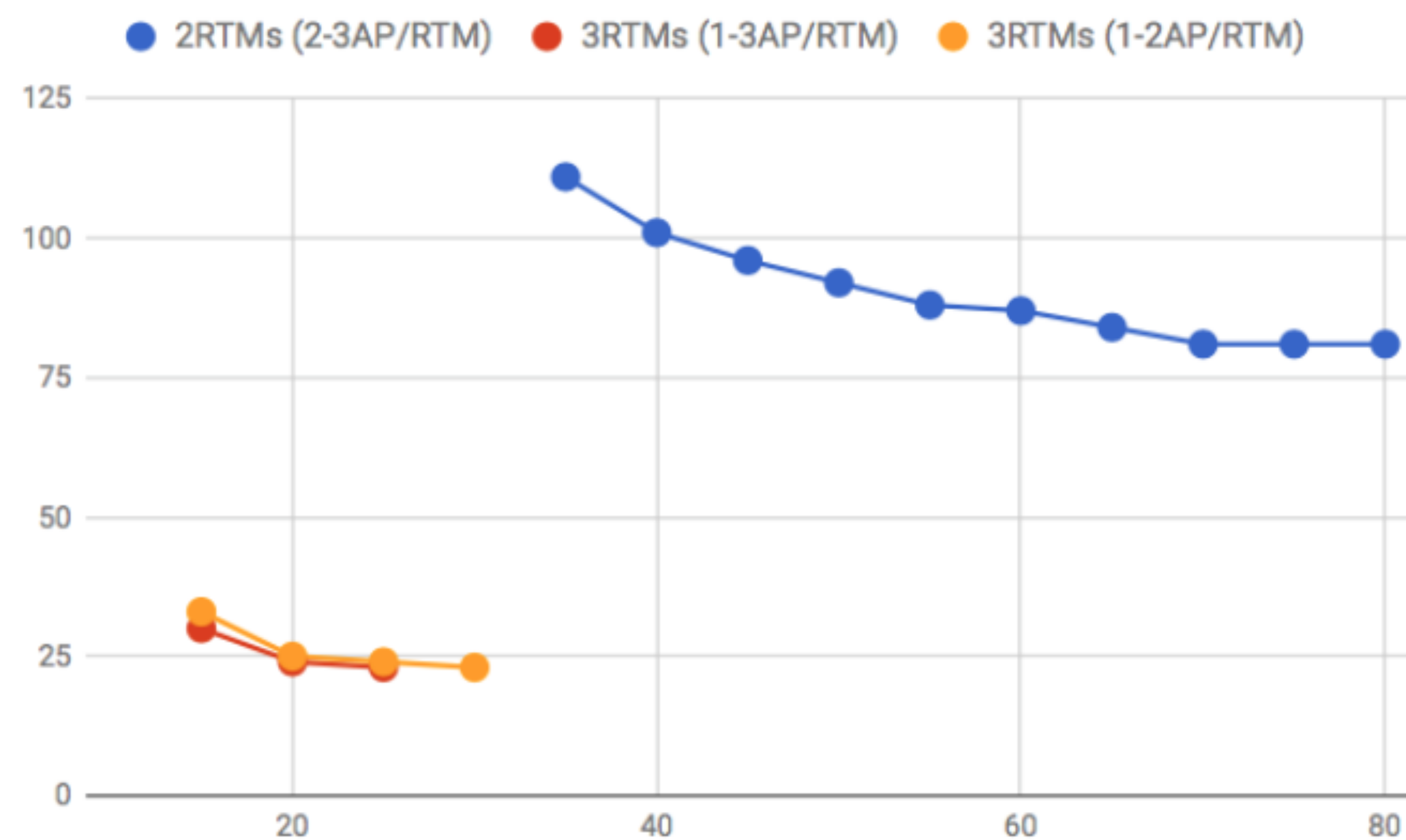
Increased Traffic Volume

δ	# of modules	S	Δ	S for 3RTMs (1-3AP/RTM)	S for 3RTMs (1-2AP/RTM)
0	5	0	-	-	-
1	3	30	5	30	33
2	3	24	10	24	25
3	3	23	15	23	24
4	3	23	20	-	23
5	2	111	25	-	-
6	2	101	30	-	-
7	2	96	35	-	-
8	2	92	40	-	-
9	2	88	45	-	-
10	2	87	50	-	-
11	2	84	55	-	-
12	2	81	60	-	-
13	2	81	65	-	-
14	2	81	70	-	-
15	2	81	75	-	-
16	2	80	80	-	-



Increased Traffic Volume

δ	# of modules	S	Δ	S for 3RTMs (1-3AP/RTM)	S for 3RTMs (1-2AP/RTM)
0	5	0	-	-	-
1	3	30	5	30	33
2	3	24	10	24	25
3	3	23	15	23	24
4	3	23	20	-	23
5	2	111	25	-	-
6	2	101	30	-	-
7	2	96	35	-	-
8	2	92	40	-	-
9	2	88	45	-	-
10	2	87	50	-	-
11	2	84	55	-	-
12	2	81	60	-	-
13	2	81	65	-	-
14	2	81	70	-	-
15	2	81	75	-	-
16	2	80	80	-	-

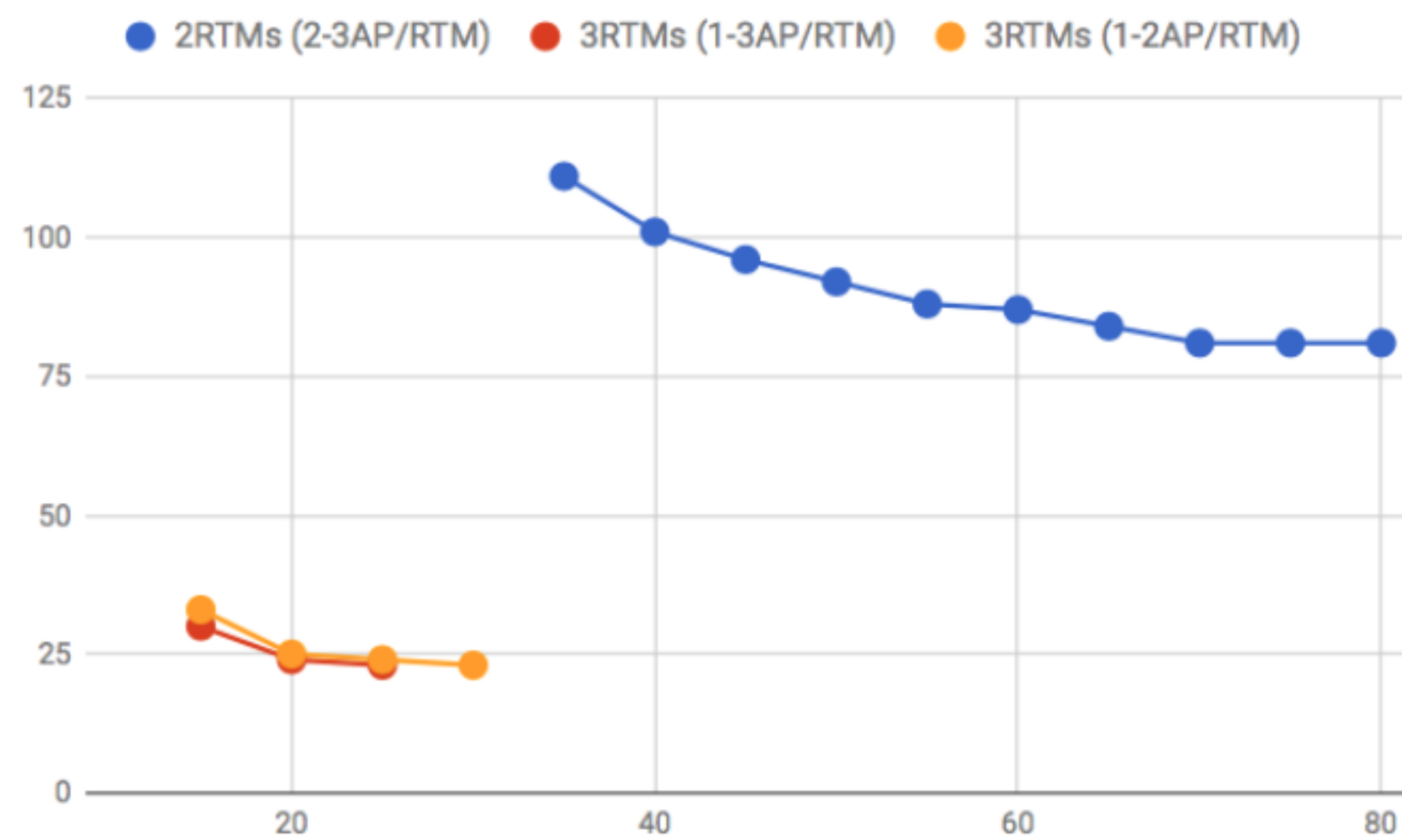


For MAP=2 we get the optimum of 3RTMs for δ=1
33 shifts ↔ 7 shifts for original traffic

Increased Traffic Volume

Same tradeoffs: more shifts — larger shifts (more minutes) — more APs/module

δ	# of modules	S	Δ	S for 3RTMs (1-3AP/RTM)	S for 3RTMs (1-2AP/RTM)
0	5	0	-	-	-
1	3	30	5	30	33
2	3	24	10	24	25
3	3	23	15	23	24
4	3	23	20	-	23
5	2	111	25	-	-
6	2	101	30	-	-
7	2	96	35	-	-
8	2	92	40	-	-
9	2	88	45	-	-
10	2	87	50	-	-
11	2	84	55	-	-
12	2	81	60	-	-
13	2	81	65	-	-
14	2	81	70	-	-
15	2	81	75	-	-
16	2	80	80	-	-



For MAP=2 we get the optimum of 3RTMs for δ=1
33 shifts ↔ 7 shifts for original traffic

Conclusion/Future Work

Conclusion

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach

➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach

➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module

➡ Minor shifts (few minutes) can significantly reduce necessary number of modules

Future Work

LINKÖPING UNIVERSITY

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach

➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module

➡ Minor shifts (few minutes) can significantly reduce necessary number of modules

➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach

➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module

➡ Minor shifts (few minutes) can significantly reduce necessary number of modules

➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach

➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module

➡ Minor shifts (few minutes) can significantly reduce necessary number of modules

➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations
➡ Possibly: distinguish arrival/departures

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations
➡ Possibly: distinguish arrival/departures
➡ Possibly: consider uncertainty

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations
➡ Possibly: distinguish arrival/departures
➡ Possibly: consider uncertainty

- Computational complexity of FRAMA with $\Delta > 0$ and even MAP=2 is open

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations
➡ Possibly: distinguish arrival/departures
➡ Possibly: consider uncertainty

- Computational complexity of FRAMA with $\Delta > 0$ and even MAP=2 is open
- Currently we do not care which airlines affected by shift (possibly all to a single airline)

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations
➡ Possibly: distinguish arrival/departures
➡ Possibly: consider uncertainty

- Computational complexity of FRAMA with $\Delta > 0$ and even MAP=2 is open
- Currently we do not care which airlines affected by shift (possibly all to a single airline)

➡ Take equity into account (2 airlines, airline A operating 150 flights, airline B operating 75; reassign slot for 60 flights→ aim for 40 new slots for airline A, 20 new slots for airline B)

Conclusion

- Optimization problem for remote towers (FRAMA):
  - Shifts flights to other, nearby, slots
  - To minimize the total number of modules in the RTC
- Discussed computational complexity
- Presented different solution approaches
- Experiments for IP for five Swedish airports

➡ Show applicability of our approach
➡ Tradeoffs: more shifts — larger shifts (more minutes) — more APs/module
➡ Minor shifts (few minutes) can significantly reduce necessary number of modules
➡ Cooperation between airlines, airport owners and ANSPs may help in reduction of RTC operation costs

Future Work

Thank you.

- Our conflict definition may be too conservative/precautionary
- They cannot be discarded, and will influence staff planning

➡ Continues discussion with operations
➡ Possibly: distinguish arrival/departures
➡ Possibly: consider uncertainty

- Computational complexity of FRAMA with $\Delta > 0$ and even MAP=2 is open
- Currently we do not care which airlines affected by shift (possibly all to a single airline)

➡ Take equity into account (2 airlines, airline A operating 150 flights, airline B operating 75; reassign slot for 60 flights → aim for 40 new slots for airline A, 20 new slots for airline B)