Relational Time-Space Data Structure To Speed Up Conflict Detection Under Heavy Traffic Conditions

Sergio Ruiz Navarro

Director of research: Dr. M.A. Piera

Universitat Autònoma de Barcelona

UAB
Universitat Autònoma de Barcelona

Outline

- Introduction
- Conflict Detection: State of the Art
  - Pairwise paradigm
  - Spatial Data Structures paradigm
- Improvements:
  - Relational Spatial Data Structures
  - Time-Space Data Structures
- Performance analysis
- Conclusions and Future Work

Introduction

Introduction

- Expected changes in the future ATM:
  - **Trajectory-based operations** (4D Trajectories)
  - Higher level of air **traffic** (expected x2)
  - Bigger air **sectors** (FABs)

STREAM project:
aims to fill the current existing gap between the strategic and the tactical planning in the ATM

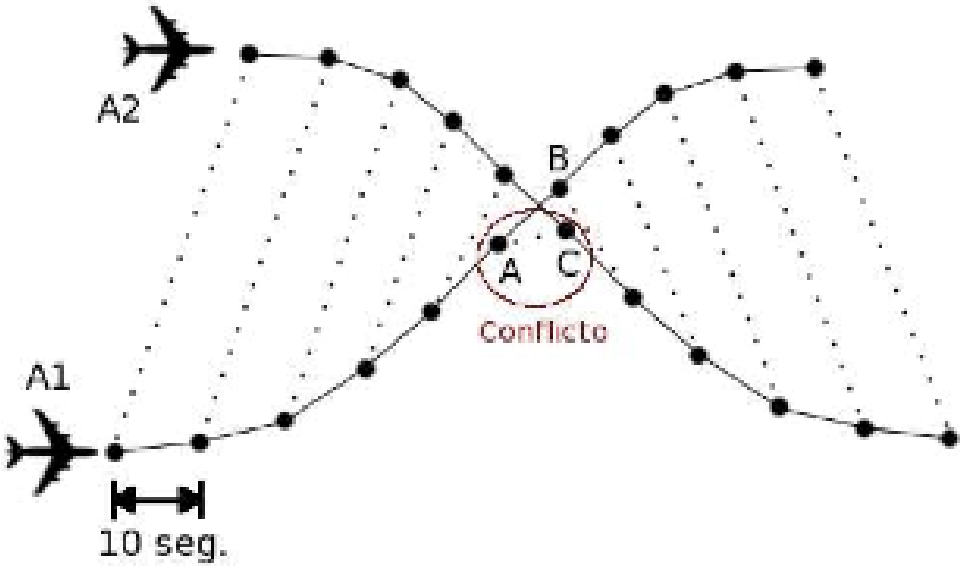
Introduction

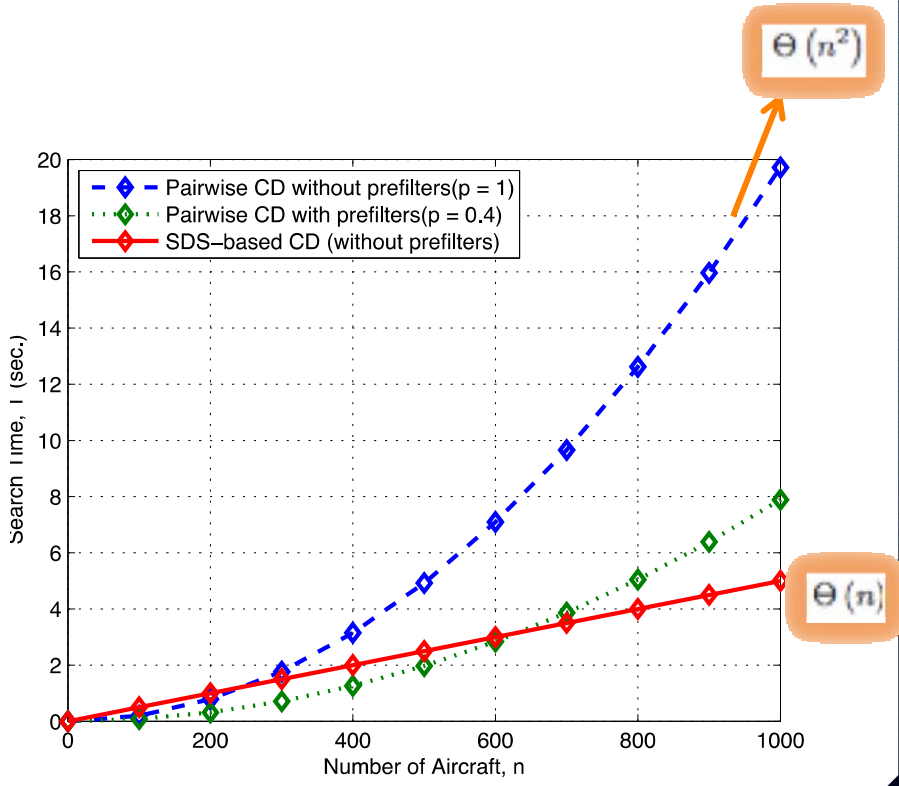
- Conflict Detection & Resolution Systems are needed to organize traffic flows in the ATM:
  - **Conflict Detection (CD)**: detect 4DT's that do not preserve *safety distances*
  - **Conflict Resolution (CR)**: recalculate a set of new conflict-free (*efficient*) trajectories

State of the Art of CD

State of the Art

- Pairwise paradigm:
  - Usually CD algorithms are based on pairwise strategies (quadratic complexity: $O(n^2)$)

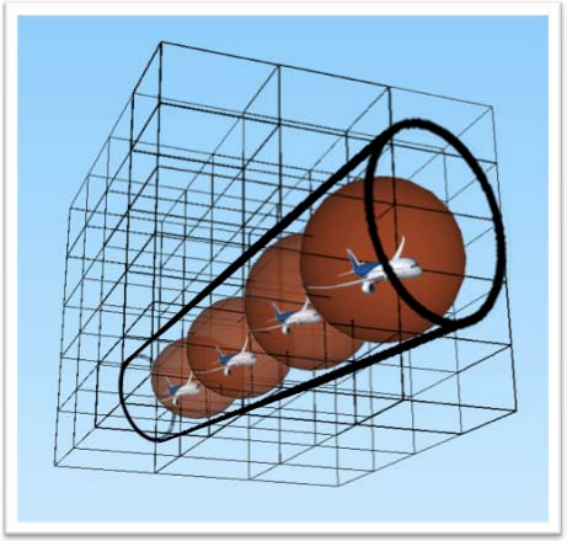




State of the Art

Paradigm based on Spatial Data Structures (SDS)

- The main idea is to collect information about a **spatial region.**
- We **discretize** the continuous space to be computer-tractable, like in a "digital snapshot" (*resolution* matters).
- Each discrete coordinate of the space can be represented by a **unique memory position** in a database
- Databases storing information sorted according to the position in the space are called **Spatial Data Structures.**
- 4^{th} dimension (**time**) can be stored inside



State of the Art

- A SDS is a **database** that represents a spatial region (e.g. an air sector) by using individual memory positions to represent each of the discrete (3D) coordinates of the sector.

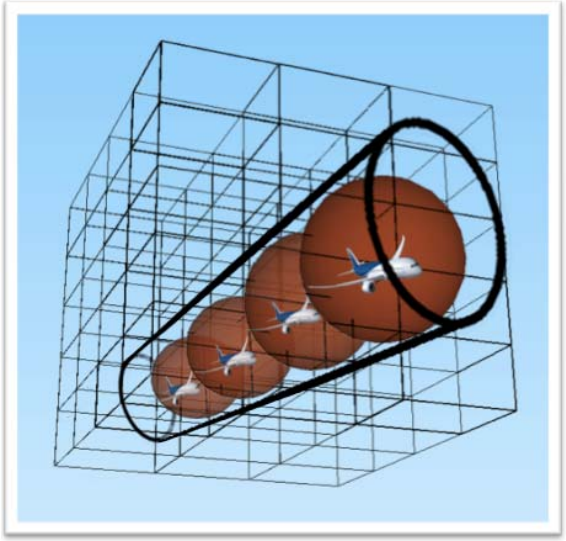
- Such memory positions are sorted in a way that, given a certain coordinate, the information stored inside the SDS is **easily recoverable** applying simple mathematical formulas:

$$SDSrow\ (x,y,z) = x \cdot Y \cdot Z + y \cdot Z + z$$

Y,Z: order or size of dimension Y and Z.

State of the Art

Physical concept of a SDS



Logical concept of a SDS

	Reservation 1			Reservation 2			Reservation 3			...	Reservation n		
Coords	Aircraft	TWon	TWoff	Aircraft	TWon	TWoff	Aircraft	TWon	TWoff	...	Aircraft	TWon	TWoff
(0,0,0)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,0,1)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,0,2)	15	50	170	0	0	0	0	0	0	...	0	0	0
(0,0,3)	0	0	0	4	76	196	0	0	0		0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
(0,1,0)	8	99	219	0	0	0	0	0	0	...	0	0	0
(0,1,1)	**3**	**34**	**154**	**4**	**32**	**152**	7	879	999	...	0	0	0
(0,1,2)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,1,3)	0	0	0	0	0	0	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
(0,2,0)	6	565	685	15	233	353	0	0	0	...	0	0	0
(0,2,1)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,2,2)	76	12320	12440	78	800	920	0	0	0	...	0	0	0
(0,2,3)	0	0	0	0	0	0	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
(1,0,0)	0	0	0	0	0	0	0	0	0	...	0	0	0
(1,0,1)	**4**	**590**	**710**	15	88	208	**19**	**680**	**800**	...	0	0	0
(1,0,2)	1	79	199	2	800	920	3	550	670	...	n	10098	10118
(1,0,3)	0	0	0	0	0	0	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮

Physical implementation of a SDS

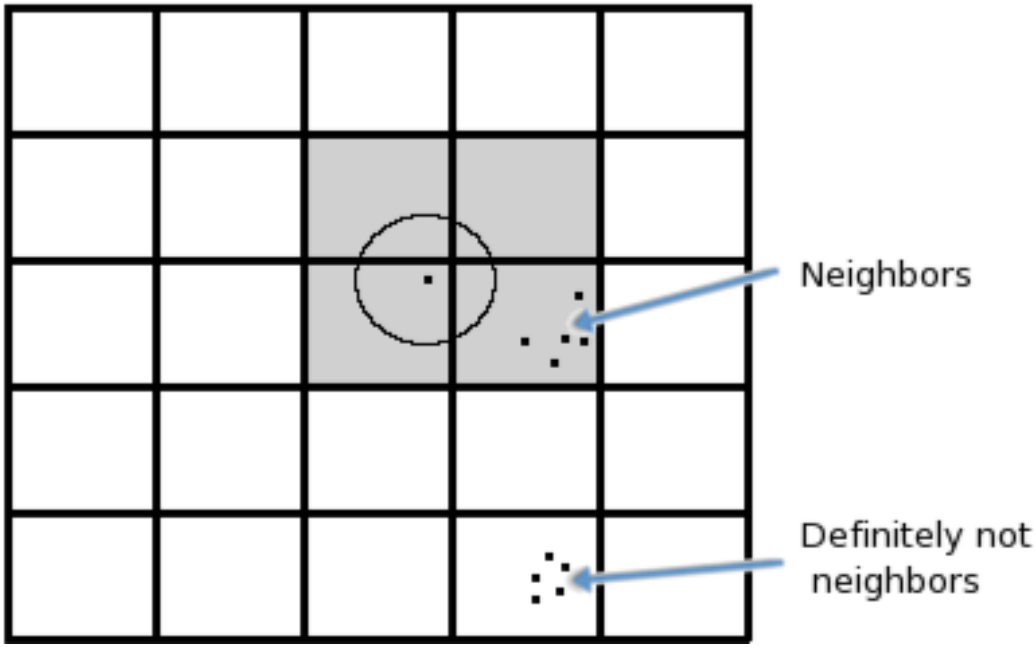
Byte 1 | Byte 2 | Byte 3 | … | Byte n

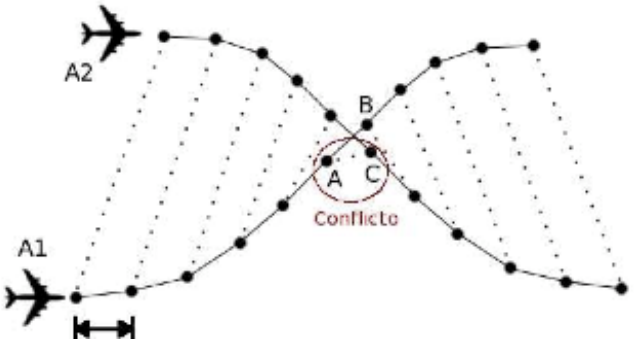
State of the Art

- 2 possible configurations of the SDS (already tested):
  - **Clustering space** for reducing Pairwise comparisons
    - Developed for videogames apps (Craig Reynolds)
  - CD through **4D Tubes**
    - Developed under ATLANTIDA project

State of the Art

+ **Clustering to reduce the amount of pairwise comparisons:**





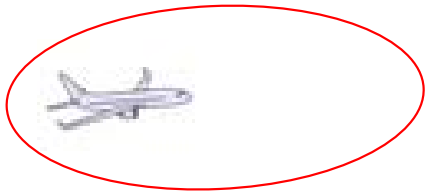
Pairwise comparison only between neighbours

➔ It reduces the $O(n^2)$ complexity to $O(n)$

State of the Art

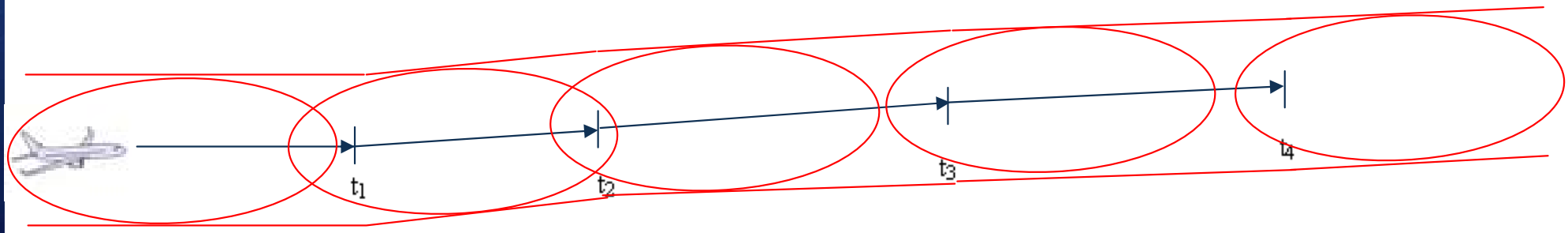
CD through 4D Tubes

- Aircraft can be modeled with a unique point in the space surrounded by a safety **ellipsoid**



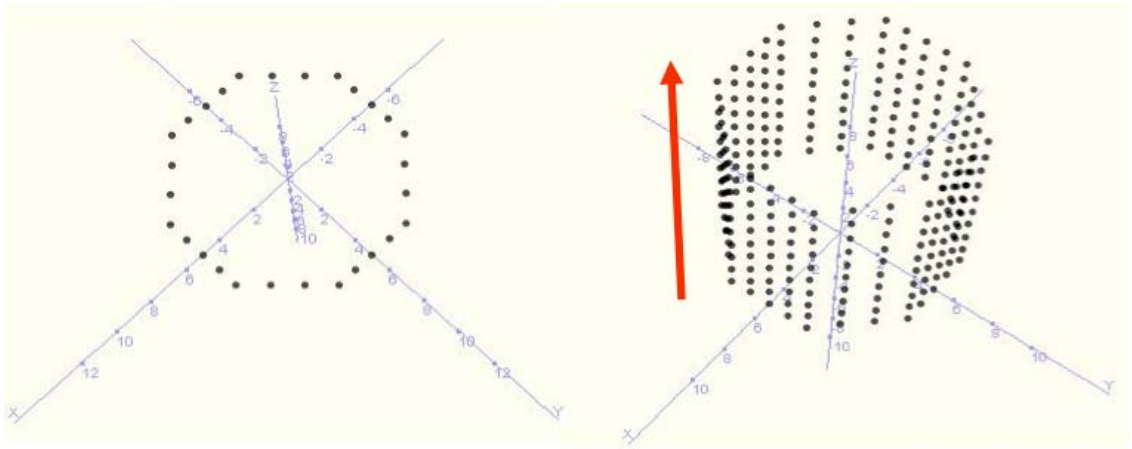
- Wake vortex can be modeled, in a given instant, as a **sphere**

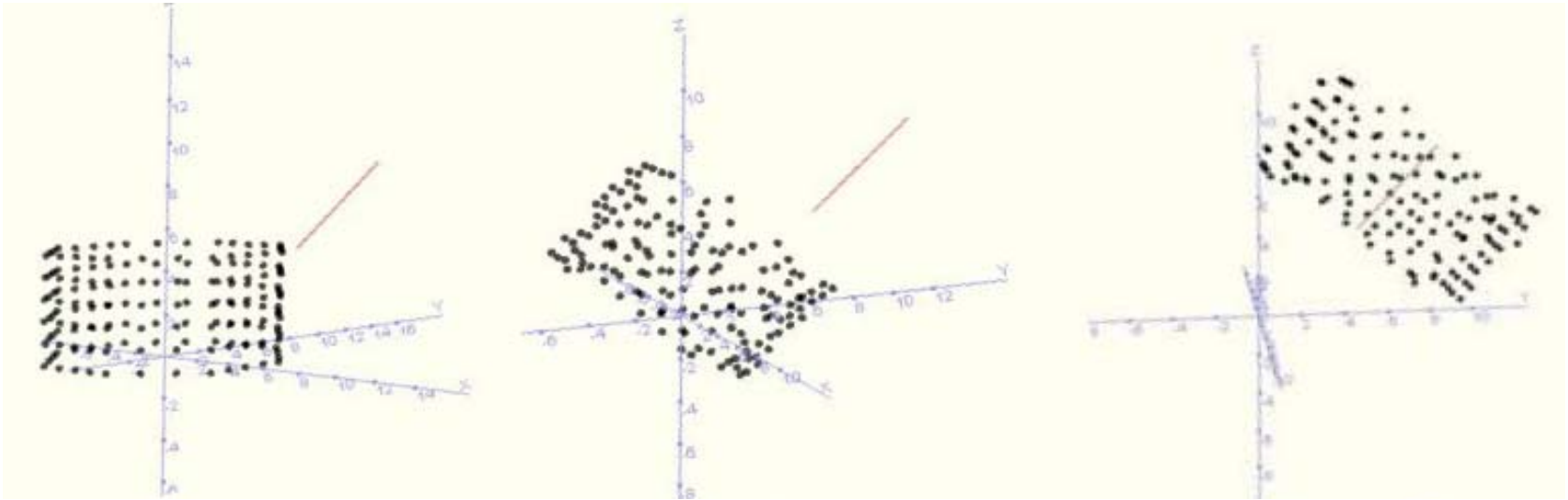
- Both ellipsoid and sphere can be thought as a **4D tube** when aircraft moves through the continuous (real) space.



State of the Art

Construction, rotation and displacement of the cylinder

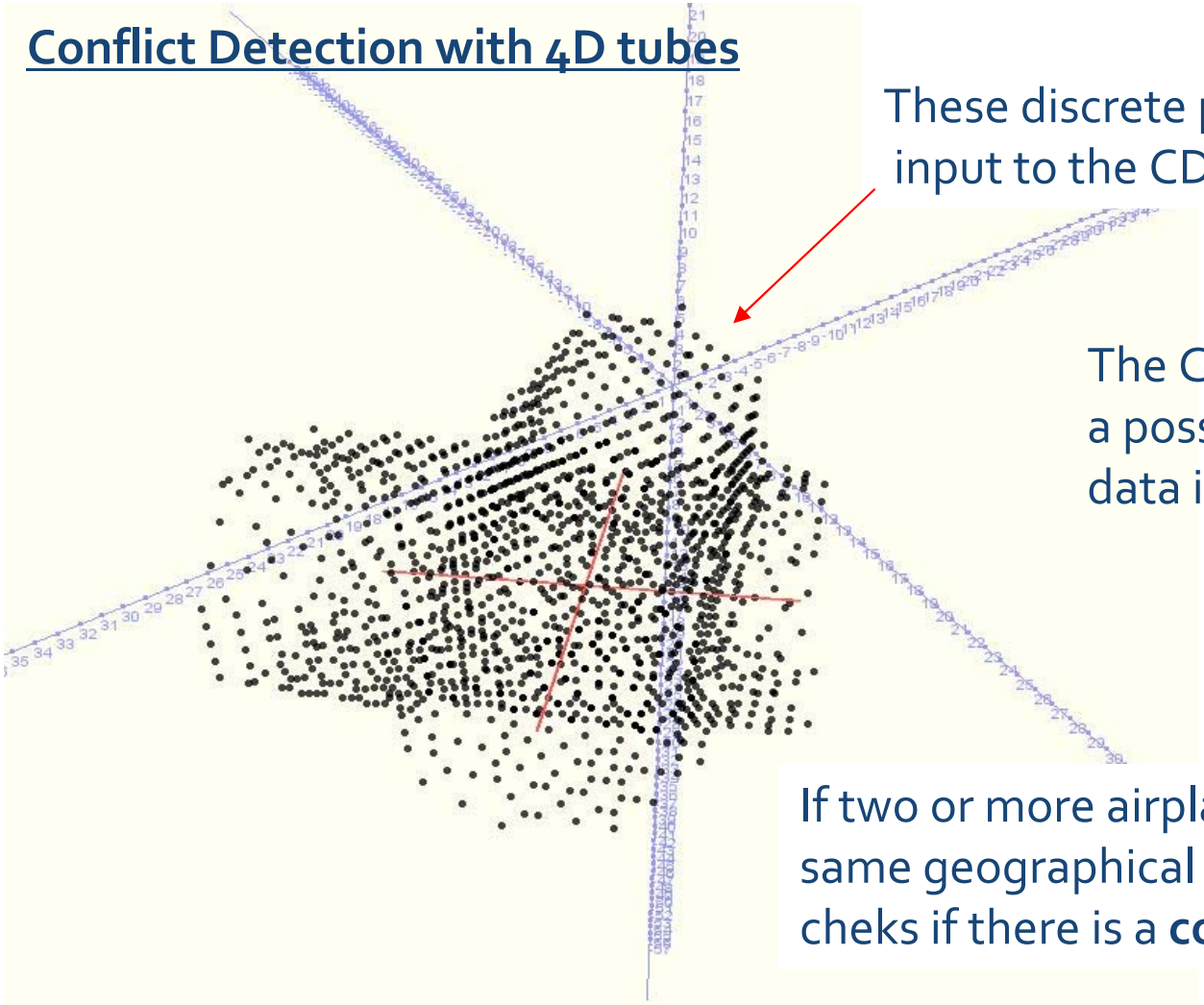




(use of Homogeneous Transformation Matrices)

State of the Art

Conflict Detection with 4D tubes



These discrete points are the input to the CD algorithm

The CD algorithm efficiently verifies a possible conflict each time a 4D data is **registered at the DB**

If two or more airplanes wants to use the same geographical 4D point, then the algorithm cheks if there is a **conflict in the time window**

State of the Art

Coords	Reservation 1			Reservation 2			Reservation 3			...	Reservation n		
	Aircraft	TWon	TWoff	Aircraft	TWon	TWoff	Aircraft	TWon	TWoff	...			
(0,0,0)	0	0	0	0	0	0	0	0	0	...			
(0,0,1)	0	0	0	0	0	0	0	0	0	...			
(0,0,2)	15	50	170	0	0	0	0	0	0	...			
(0,0,3)	0	0	0	4	76	196	0	0	0				
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮					
(0,1,0)	8	99	219	0	0	0	0	0					
(0,1,1)	**3**	**34**	**154**	**4**	**32**	**152**	7			...	0		0
(0,1,2)	0	0	0	0	0	0	0			...	0		0
(0,1,3)	0	0	0	0	0	0	0		0	...	0		0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮		⋮	...	⋮		⋮
(0,2,0)	6	565	685	15	233		0	0	0		0	0	
(0,2,1)													
(0,2,2)													
(0,2,3)													
⋮													
(1,0,0)													
(1,0,1)													
(1,0,2)													
(1,0,3)													
⋮													

Detected conflicts can be poured out to another database

	A	B	C	E	F	G	H	J	K
1	Coordinates		Conflict 0		Conflict 1				
2		Aircraft:	Tmin:		Tmin:	Tmax:	Aircraft:	Tmin:	Aircraft:
3	(x:0, y:0, z:11)	3	8		11	15			
4	(x:0, y:0, z:18)	1	3		4	7			
5	(x:0, y:0, z:23)	4	16					19	22
6									
7									
8									
9									
10									
11									
12									
13									

Or directly passed to CR to be solved

CR System

SDS content example

Improvements for the SDS-CD

Main shortage of original SDS

- We use RAM memory for efficiency purposes
- Modeling a 3D scenario requires a lot of memory
  - Example: TMA: 270 x 270 x 120 Km^3

    30 aircrafts

    4 bytes per booking

    2700*2700*120*30*4 = **105 Gbytes of RAM !!!**

Relational SDS (RSDS)

SDS (non relational)

Coords	Reservation 1			Reservation 2			Reservation 3			...	Reservation n		
	Aircraft	TWon	TWoff	Aircraft	TWon	TWoff	Aircraft	TWon	TWoff	...	Aircraft	TWon	TWoff
(0,0,0)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,0,1)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,0,2)	15	50	170	0	0	0	0	0	0	...	0	0	0
(0,0,3)	0	0	0	4	76	196	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
(0,1,0)	8	99	219	0	0	0	0	0	0	...	0	0	0
(0,1,1)	**3**	**34**	**154**	**4**	**32**	**152**	7	879	999	...	0	0	0
(0,1,2)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,1,3)	0	0	0	0	0	0	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
(0,2,0)	6	565	685	15	233	353	0	0	0	...	0	0	0
(0,2,1)	0	0	0	0	0	0	0	0	0	...	0	0	0
(0,2,2)	76	12320	12440	78	800	920	0	0	0	...	0	0	0
(0,2,3)	0	0	0	0	0	0	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
(1,0,0)	0	0	0	0	0	0	0	0	0	...	0	0	0
(1,0,1)	**4**	**590**	**710**	15	88	208	**19**	**680**	**800**	...	0	0	0
(1,0,2)	1	79	199	2	800	920	3	550	670	...	n	10098	10118
(1,0,3)	0	0	0	0	0	0	0	0	0	...	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮

➔Important reductions of the memory needs using RSDS (about **98% less memory**)

RSDS (relational)

Base-SDS (BS)

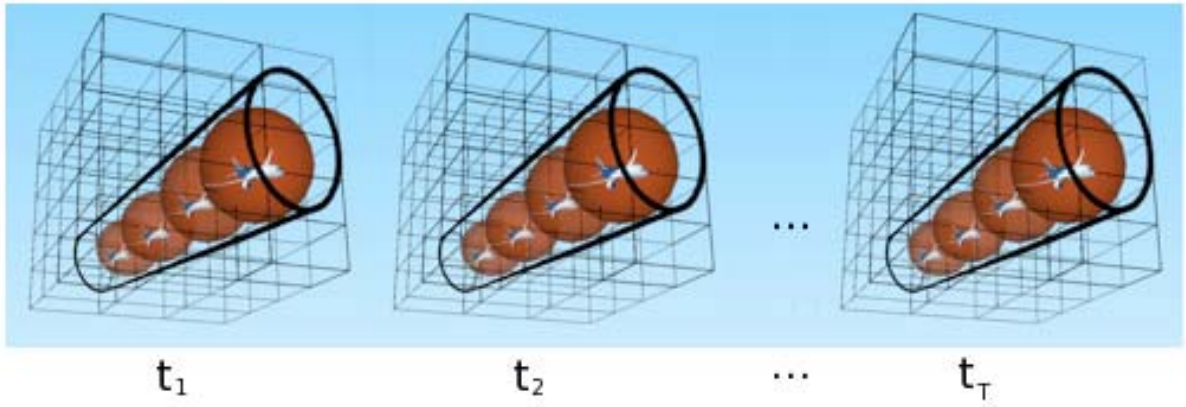
Coords	Pointer to a STI position
(0,0,0)	0
(0,0,1)	0
(0,0,2)	12295
(0,0,3)	0
(0,1,0)	530
(0,1,1)	5
(0,1,2)	0
(0,1,3)	0
(0,2,0)	43
(0,2,1)	0
(0,2,2)	12
(0,2,3)	0
(0,2,2)	0
(0,2,3)	1000443
(0,3,0)	0
(0,3,1)	0
(0,3,2)	63
(0,3,3)	969
(1,0,0)	0

Stacked Trajectory Information (STI)

Position	Aircraft ID	Twmin	Twmax	Pointer to a STI position
1	4	5	125	0
2	2	6565	6685	4234
3	2	2534	2654	54
4	4	76	196	764
5	3	34	154	10
6	3	2565	2685	75
7	67	753	873	99
8	8	56313	56433	456
9	6	565	685	0
10	4	32	152	14
11	76	56567	56687	786
12	345	2346	2466	0
13	6	7	127	0
14	7	879	999	0
15	9	9976	10096	0
16	1	6	126	867

Time-Space Data Structure

- Now each row of the Data Structure represents a time-spatial region (i.e. a 4D resource)



- The access to the position is calculated by:

$$SDSrow(x,y,z,t) = x \cdot Y \cdot Z \cdot T + y \cdot Z \cdot T + z \cdot T + t$$

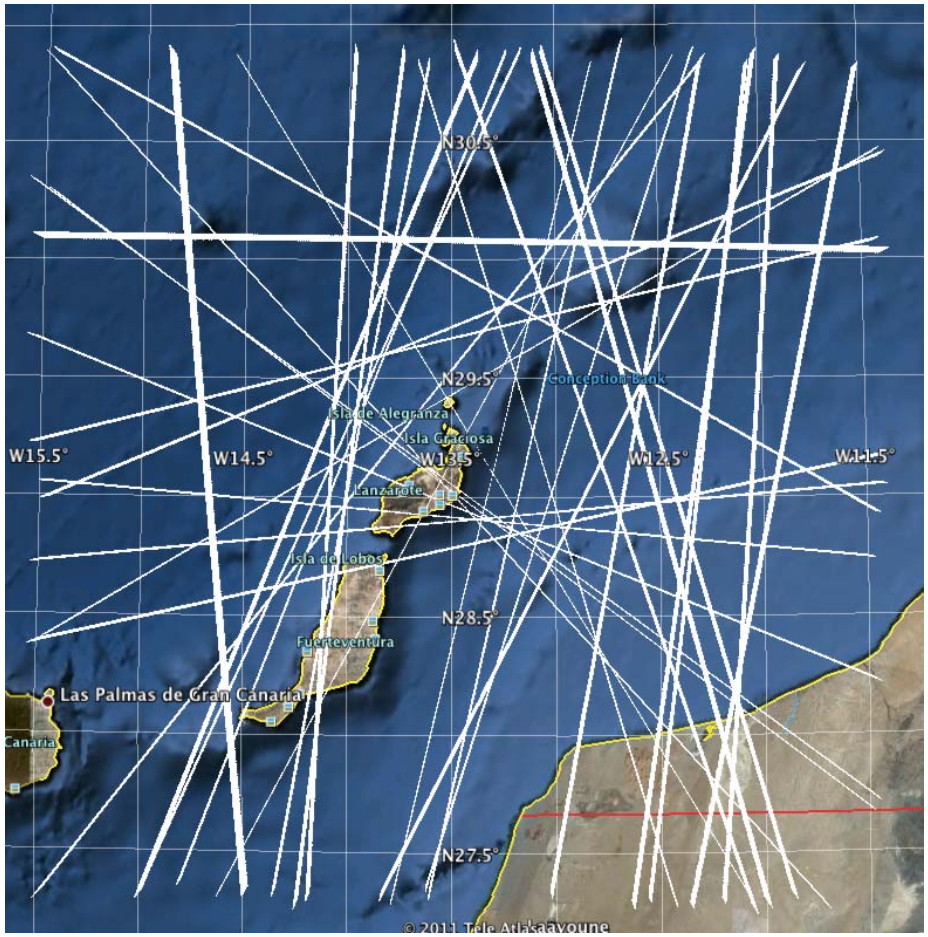
- RSDS + TSDS = RTSDS (it is possible to join the concepts)

Performance of RTSDS

Performance of RTSDS

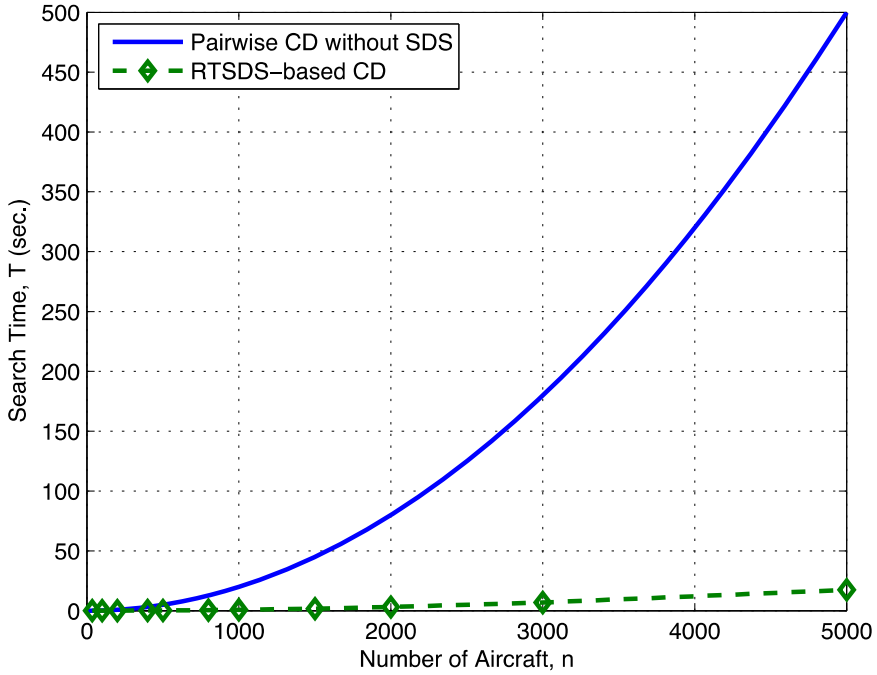
- Scenario:
  - Configuration: pairwise pruning
  - 400x400Km2
  - From 30 to 5000 4DTs
  - ½ hour duration each 4DT
  - Horizontal discr.: 20Km
  - Vertical discr.: 20FL
  - Temporal resol.: 1 sec.

Use of memory < 1GB



Performance of RTSDS

n	Pairwise T [ms]	RTSDS T [ms]
35	23	22
100	194	52
200	785	97
400	3153	213
500	4925	287
800	12620	579
1000	19738	837
1500	44447	1675
2000	79074	3200
3000	178008	6837
5000	493802	17513



With less congestion the performance would be better

Performance of RTSDS

- Storage of the State-Space allows a very good integration with CR block:
  - Easy to detect conflicts between original trajectories and alternatives trajectories, and also between 2 alternative trajectories.
  - Could reduce the complexity of CR algorithms.
  - Enable the design of CR that explores the state-space (e.g. Coloured Petri Nets).

Conclusions

Conclusions

- Pairwise-based CD algorithms, currently widely used on major real CD systems, may be improved by using of **SDS-based CD algorithms**, since:
  - they **run at linear time** (linear temporal complexity, O(n))
  - they **store the state-space** of the problem, which may help CR systems (global strategy).
- **Relational SDS** has been designed to reduce the exponential growth of memory up to 95-99% of the original **RAM memory requirements**.
- **Time-Space Data Structure** introduce a new concept of SDSs by introducing a temporal dimension in the data structure.
- Simulations with congested scenarios have illustrated that a combination of RSDS and TSDS, called **RTSDS, has an excellent performance.**

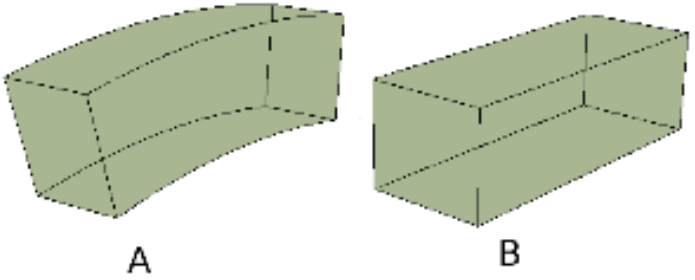
Future Work

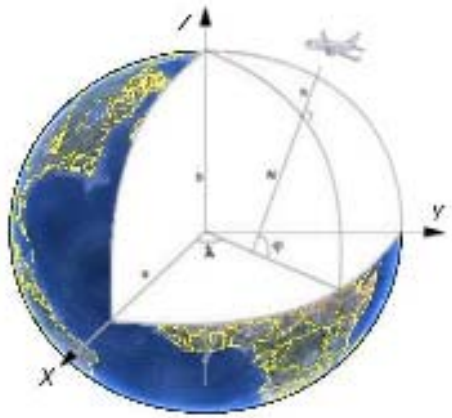
Future Research

- Add *en-route* real traffic and consider **strategic and tactical CD&R** (STREAM project)
- Integrate the SDS with **Coloured Petri Nets** to take advantage of the stored state-space (optimality).
- Add weather perturbations and more precise **vortex modeling** (WAKE4D project and FLYSAFE project)
- Determine how to deal with errors due to Earth **planar projection in big sectors** (i.e. FABs)

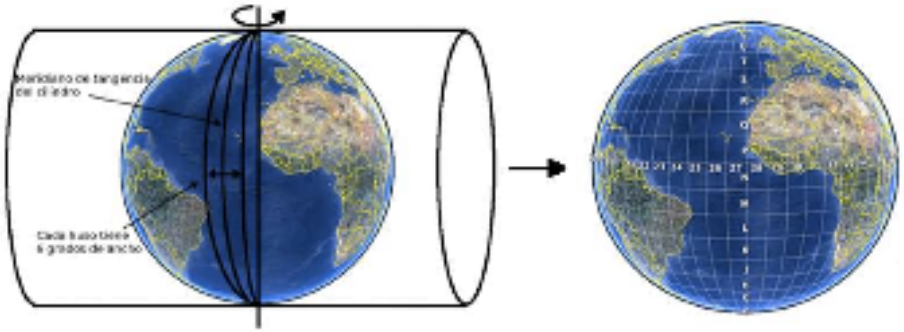
CD algorithm

- It should be taken into account the curvature of Earth





- UTM projection allows making a planar projection with very few distortion



- Two conversions: WGS-84 → UTM and after UTM → WGS-84

Thank you for your attention!

- QUESTIONS?