Around Quantum Decision Diagrams

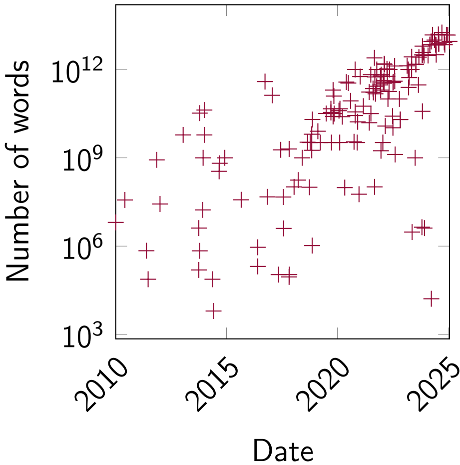
Malo Leroy

Research Track – CentraleSupélec

February 17, 2025

Databases grow rapidly in size

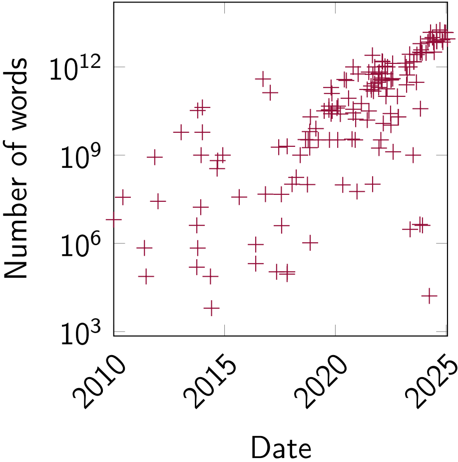
Classical algorithms are sometimes not efficient enough



Number of words used to train language models

Databases grow rapidly in size

Classical algorithms are sometimes not efficient enough



Quantum algorithms enable us to solve some probels faster

Number of words used to train language models

Quantum machines are still in development and will remain expensive

$$\Downarrow$$

There is a need for quantum algorithm **simulation** and verification tools

Simulations are very costly in terms of computation time

Grover	Classical	Quantum	Simulation
Complexity	N	$\sqrt{N}$	$N\sqrt{N}$

Simulation requires adapted **data structures**

State of the art

- Abstract interpretation
- Arithmetics of real intervals
- Quantum decision diagrams

State of the art

- Abstract interpretation
- Arithmetics of real intervals
- Quantum decision diagrams

Solution : abstract additive decision diagrams

Abstract interpretation enables us to determine mathematical properties or to speed up calculations

Example : sign of an expression $e = (3 + 2) \times (-5)$

$$\begin{aligned} \mathrm{signe}(e) &= (\mathrm{signe}(3) + \mathrm{signe}(2)) \times \mathrm{signe}(-5) \\ &= (\oplus + \oplus) \times \ominus \\ &= \oplus \times \ominus \\ &= \ominus \end{aligned}$$

Abstract interpretation enables us to determine mathematical properties or to **speed up calculations**

It can be exact or **approximate**

Abstract interpretation can be applied to **real intervals**

$$
\begin{aligned}
[1,2] * [-1,1] &= [-2,2] \\
[1,2] + [-1,1] &= [0,3] \\
[1,2] \wedge [-1,1] &= [1,1]
\end{aligned}
$$

The result of the operation is **the smallest interval** containing all the possible 1-to-1 results

Una **Boolean function**

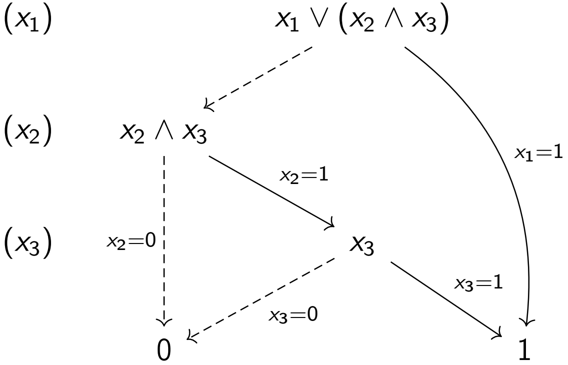
$$f : \{0,1\}^n \to \{0,1\}$$

can be represented by a **truth table**

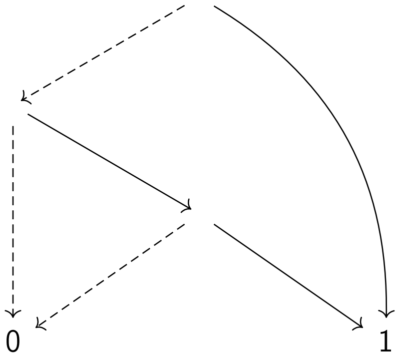
x_1	x_2	x_3	$f(x_1, x_2)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

for $f(x_1, x_2) = x_1 \vee (x_2 \wedge x_3)$

Decision diagrams can represent Boolean functions



Decision diagrams can represent Boolean functions



We make use of the internal **structure** of the function

A **quantum state** is a superposition of incompatible states

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \quad \text{(un qubit)}$$

A **quantum state** is a superposition of incompatible states

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \quad \text{(un qubit)}$$

$$n \text{ qubits} \Rightarrow 2^n \text{ incompatible states}$$

The states are noted in the form of **vecteurs**

$$\alpha\,|01\rangle + \beta\,|10\rangle = \begin{pmatrix} \alpha \\ 0 \\ \beta \\ 0 \end{pmatrix}$$

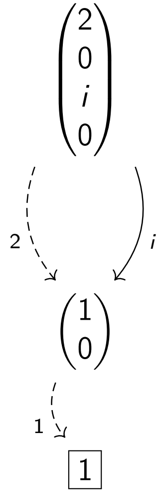
The usual representation is similar to **truth tables**

x_1	x_2	$\langle x_1 x_2 \vert \psi \rangle$
0	0	α
0	1	0
1	0	β
1	1	0

for $|\psi\rangle = \alpha\,|00\rangle + \beta\,|10\rangle$

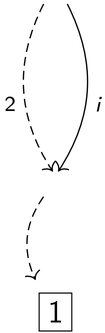
States can be represented by **quantum decision diagrams**

We make use of the internal **structure** of the state



States can be represented by **quantum decision diagrams**

In the worst case it is still space-**exponential** in n



Look back at the state of the art

- ✓ Abstract interpretation
- ✓ Arithmetics of real intervals
- ✓ Quantum decision diagrams

We will use these concepts together,
with an innovation : l'**additivity**

$$\begin{tikzcd} & \Psi \arrow[dl, dashed] \arrow[d, dashed] \arrow[dr] & \\ \phi_1 & \phi_2 & \psi \end{tikzcd} \quad \Leftrightarrow \quad \begin{tikzcd} & \Psi \arrow[dl, dashed] \arrow[dr] & \\ \phi_1 + \phi_2 & & \psi \end{tikzcd}$$

Look back at the state of the art

- ✓ Abstract interpretation
- ✓ Arithmetics of real intervals
- ✓ Quantum decision diagrams
- + Innovation : additivity

Solution : abstract additive decision diagrams

Objectives

- **Formal model** for abstract additive quantum decision diagrams
- **Implementation** of the model

Methodology

$$\begin{tikzcd}
{} \arrow[r] & \text{Documentation} \arrow[rr, "\text{usual cases}"] \arrow[dr, "\text{state of the art}"'] & & \text{Examples} \arrow[dl, bend left, "\text{model}"'] \arrow[dr, "\text{tests}"] & \\
& & \text{Theory} \arrow[ur, bend left] \arrow[rr, "\text{code}"] & & \text{Code}
\end{tikzcd}$$

GitHub for project management

- **Issues** for tasks and bugs
- Priorities, sizes, and deadlines
- Branches and **merge requests**

Model

- S6 Cartesian & polar intervals of $\mathbb{C}$
- S6 Diagrams
- S6 Local and global approximations
- S6 Forced merge
- S6 Redcution algorithms
- S7 Error
- S7 Gate application

Example : we consider the state $\begin{pmatrix} 10i+2 \\ 4i+1 \\ 2i \\ i \end{pmatrix}$

Example : we consider the state $\begin{pmatrix} 10i + 2 \\ 4i + 1 \\ 2i \\ i \end{pmatrix}$

There are **regularities**.

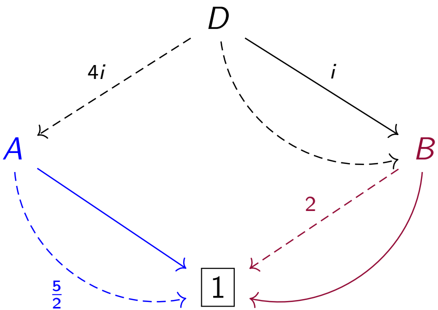
Example : we consider the state $\begin{pmatrix} \begin{pmatrix} 10i \\ 4i \end{pmatrix} + \begin{pmatrix} 2 \\ 1 \end{pmatrix} \\ i \begin{pmatrix} 2 \\ 1 \end{pmatrix} \end{pmatrix}$

There are **regularities**.

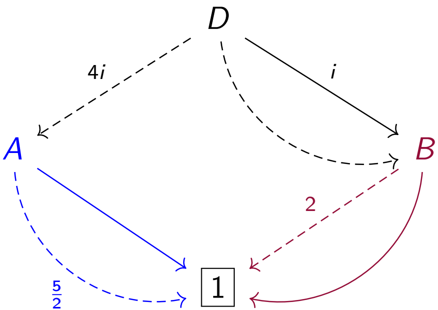
Example : we consider the state $\begin{pmatrix} 4i \begin{pmatrix} 5/2 \\ 1 \end{pmatrix} + \begin{pmatrix} 2 \\ 1 \end{pmatrix} \\ i \begin{pmatrix} 2 \\ 1 \end{pmatrix} \end{pmatrix}$

There are **regularities**.

Example : we get the additive diagram

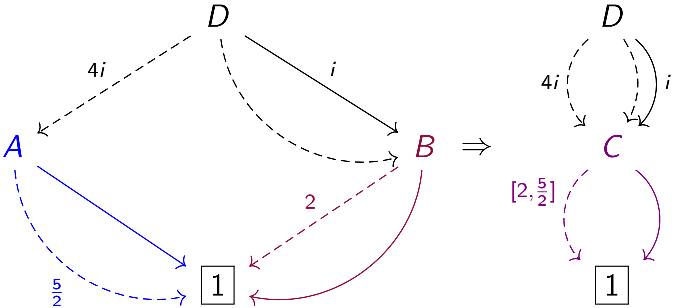


Example : we get the additive diagram

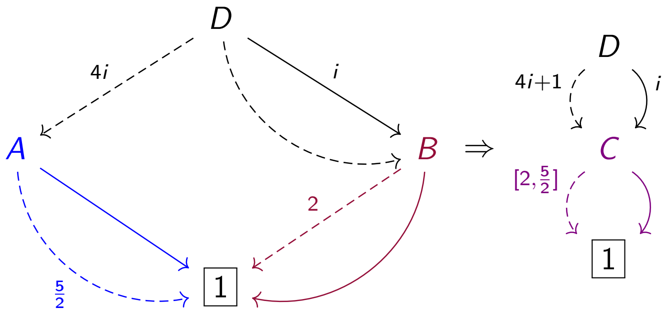


Let's **reduce** this diagram

We can force the merge of A and B



We can force the merge of A and B



We can always reduce diagrams more

$$\Downarrow$$

Arbitrarily large space gain (up to exponential)

How to choose which diagrams to merge?

$$\Downarrow$$

Error : we can compute the error induced by a diagram

Quantum circuits are based on **gates**

They are represented by **matrices**

Quantum circuits are based on **gates**

They are represented by **matrices**

Example : Hadamard

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$H \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

To apply a gate M to a diagram D

To apply a gate M to a diagram D

If $\mathcal{E}(D) =$ evaluation of D

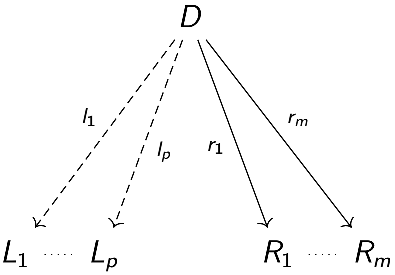
We *want* $\mathcal{E}(M(D)) = M\ \mathcal{E}(D)$

To apply a gate M to a diagram D

If $\mathcal{E}(D) =$ evaluation of D

We *want* $\mathcal{E}(M(D)) = M\ \mathcal{E}(D)$

$$D = (\{(l_1, L_1), ..., (l_p, L_p)\}, \{(r_1, R_1), ..., (r_m, R_m)\})$$

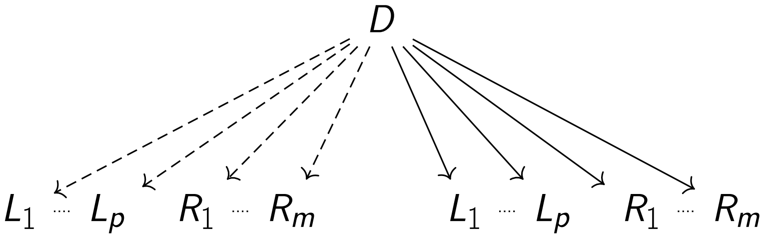


Before application of M

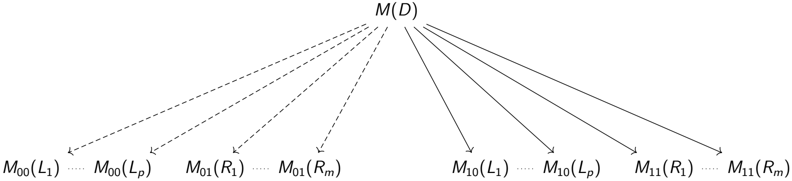
1.

$$\begin{tikzcd}
 & & & & & & M(D) \arrow[dllllll, dashed] \arrow[dllll, dashed] \arrow[dlll, dashed] \arrow[dl, dashed] \arrow[dr] \arrow[drrr] & & & \\
L_1 & \cdots & L_p & R_1 & \cdots & R_m & & R_1 & \cdots & R_m
\end{tikzcd}$$

2.



3.



$$M = \begin{pmatrix} M_{00} & M_{01} \\ M_{10} & M_{11} \end{pmatrix}$$

3.

$$
\begin{tikzcd}[column sep=small]
 & & & & & & M(D) \arrow[dllllll, dashed] \arrow[dllll, dashed] \arrow[dlll, dashed] \arrow[dl, dashed] \arrow[d] \arrow[drr] \arrow[drrr] \arrow[drrrrr] & & & & & \\
M_{00}(L_1) & \cdots & M_{00}(L_p) & M_{01}(R_1) & \cdots & M_{01}(R_m) & M_{10}(L_1) & \cdots & M_{10}(L_p) & M_{11}(R_1) & \cdots & M_{11}(R_m)
\end{tikzcd}
$$

$$\mathcal{E}(M(D)) = \begin{pmatrix} \sum l_i M_{00}\mathcal{E}(L_i) + \sum r_j M_{01}\mathcal{E}(R_j) \\ \sum l_i M_{10}\mathcal{E}(L_i) + \sum r_j M_{11}\mathcal{E}(R_j) \end{pmatrix} = M\ \mathcal{E}(D)$$

Implementation

- S6 Cartesian & polar intervals of $\mathbb{C}$
- S6 Diagrams : building, evaluation
- S6 Forced merge
- S6 Redcution algorithms
- S7 Random diagrams
- S7 Error
- S7 Gate application
- S7 QASM

Future developments

- **Implementation**
  - Graphical interface
  - Benchmarks
- **Tweaks**
  - Error function
  - Reduction algorithms
- New concepts
  - Tree automata
  - Local Invertible Map Decision Diagrams (LIMDD)

Project framework
Future training

- **Supervisor** : Renaud Vilmart
- **Team** : QuaCS
- **Laboratory** : Laboratoire Méthodes Formelles

LMF Laboratoire Méthodes Formelles

QuaCS

Gap year *Digital Tech Year*

- Semester at Paris Digital Lab
- Various tech projects in teams
- 6-month internship in a company or lab, in France or abroad

Gap year *Digital Tech Year*

- Semester at Paris Digital Lab
- Various tech projects in teams
- 6-month internship in a company or lab, in France or abroad

After the gap year

- S8 at CentraleSupélec
- S8 Pro (internship)
- Academic S8 abroad

Majors / mentions

- **Computer science and digital technology**
  - Software engineering
  - Computer systems architecture
- **Physics and nanotechnologies**
  - Quantum engineering

Other training programs

- ARTeQ (ENS Paris-Saclay)
- QMI M2 (Télécom Paris, among others)

Conclusion

Questions

Implementation

- **Code** (4.9k lines)
  - Langage C++
  - LLVM / Clang
  - Ninja
  - CMake
- **Tests**
  - Google Test
  - GitHub Actions

Tools

- **Version control**
  - Git
  - GitHub
- **Documentation**
  Doxygen

$$\rho(\boxed{1}) = \{1\}$$

$$\varepsilon(\boxed{1}) = \{0\})$$

$$\forall G, D \in \mathscr{P}_f(\mathcal{A}_0 \times \mathcal{D}_n), \rho((G, D)) = \left(\sum_{(l,L) \in G} l\rho(L) \right) \bigsqcup \left(\sum_{(r,R) \in D} r\rho(I$$

$$\forall G, D \in \mathscr{P}_f(\mathcal{A}_0 \times \mathcal{D}_n),$$

$$\varepsilon((G, D)) = \left(\sum_{(l,L) \in G} l \max |\rho(L) \ominus \varepsilon(L)| + \varepsilon(L) \right) \bigsqcup \left(\sum_{(r,R) \in D} r \max |$$