



Big Data

Performance, efficiency and scalability metrics

Stéphane Vialle & Gianluca Quercini



université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies
de l'information
et de la communication (STIC)



1

POLYTECH PARIS-SACLAY

What to do with more computer resources?

Maximal size of a processed problem
(size up)

Nb of independent problems processed per seconde
(flow strength increase)

Processing speed of a problem
(speedup)

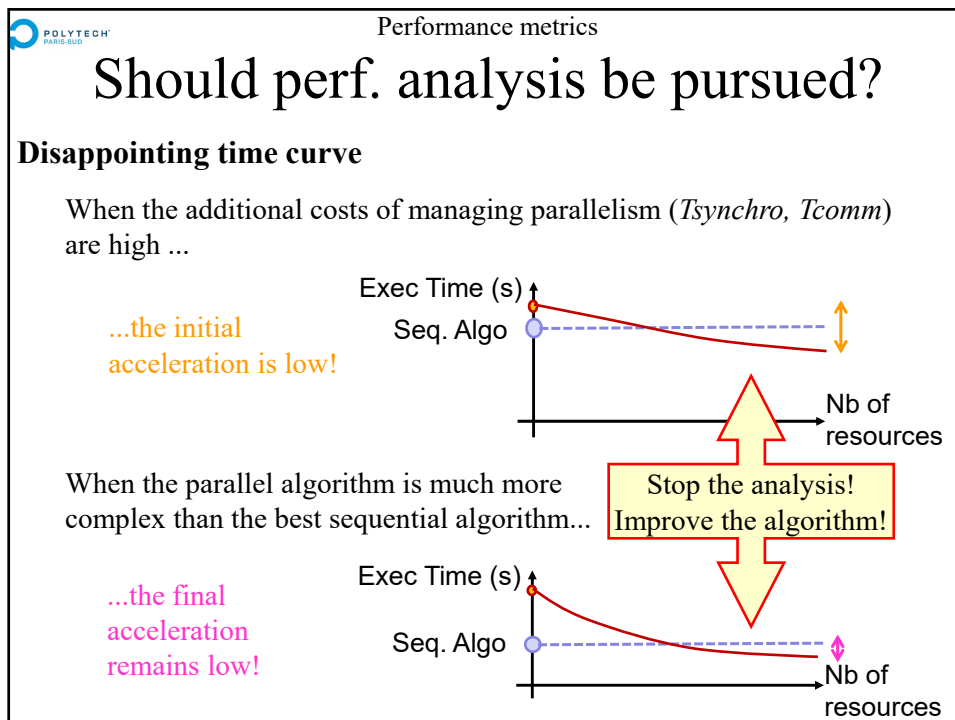
Passage à l'échelle / Scaling up : size up + speedup + cost control

2

1 – Preliminary analysis of the execution time

- Should performance analysis be pursued?
- Choice of an adapted representation

3



4

POLYTECH PARIS-SUD

Performance metrics

Should perf. analysis be pursued?

Promising time curve

When the parallel algorithm has :

- an execution time that decreases significantly
- a limited additional cost on a single resource

All accelerations will be great !

Exec Time (s)

Seq. Algo

Nb of resources

Interesting solution!
Continue the analysis

Texec vs Texec ideal
Speedup
Efficiency
Size up
Scalability...

5

POLYTECH PARIS-SUD

Performance metrics

Choice of an adapted representation

Which representation to adopt for an execution time curve?

Texec (s)

Nb of ressources

measures

What is the « best » experimental curves?

- We must compare the measurements to the theory, to the *ideal curve*
- We need a *representation* that is easy to verify *for human eyes*

Case of a parallel execution time:

- Ideal curve: $T(n \text{ ressources}) = T(1)/n$: a hyperbole
→ Check if we get a hyperbole
- But human eyes do not detect hyperboles
→ It is easily confused with a curve from another law!

6

POLYTECH PARIS-SUD

Performance metrics

Choice of an adapted representation

Which representation to adopt for an execution time curve?

measures

ideal

Nb of ressources

Texec (s)

Texec (s) (log)

Nb of ressources (log)

ideal

Pb!

Ok

Ideal curve: $T(n) = T(1)/n$

$Y = \log(T(n)) = \log(T(1)) - \log(n)$

$Y = C^{te} - X$

- With logarithmic scales, *ideal Texec* is a straight line of slope -1
- So you can easily detect:
 - a close mathematical law : $T(n) = T(1)/n^a \rightarrow$ slope $-a$
 - a complete deviation from the theory

7

POLYTECH PARIS-SUD

Performance metrics

Choice of an adapted representation

Which representation to adopt for an execution time?

Summary:

It is preferable to know the ideal/theoretical curve ...

...and to choose a representation that allows to visualize straight lines or simple geometric shapes.

ideal

Nb of ressources

Texec (s)

Texec (s) (log)

Nb of ressources (log)

ideal

Pb!

Ok

8

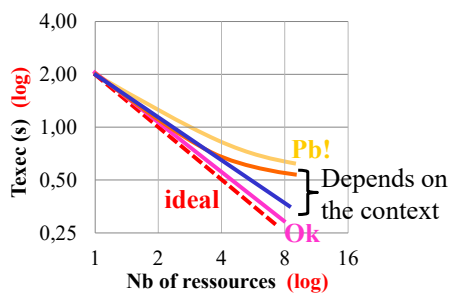
2 – Performance metrics (T, S, e)

- Execution time metrics
- Speed up metrics
- Efficiency metrics
- Sequential reference issue

9

Execution time metric

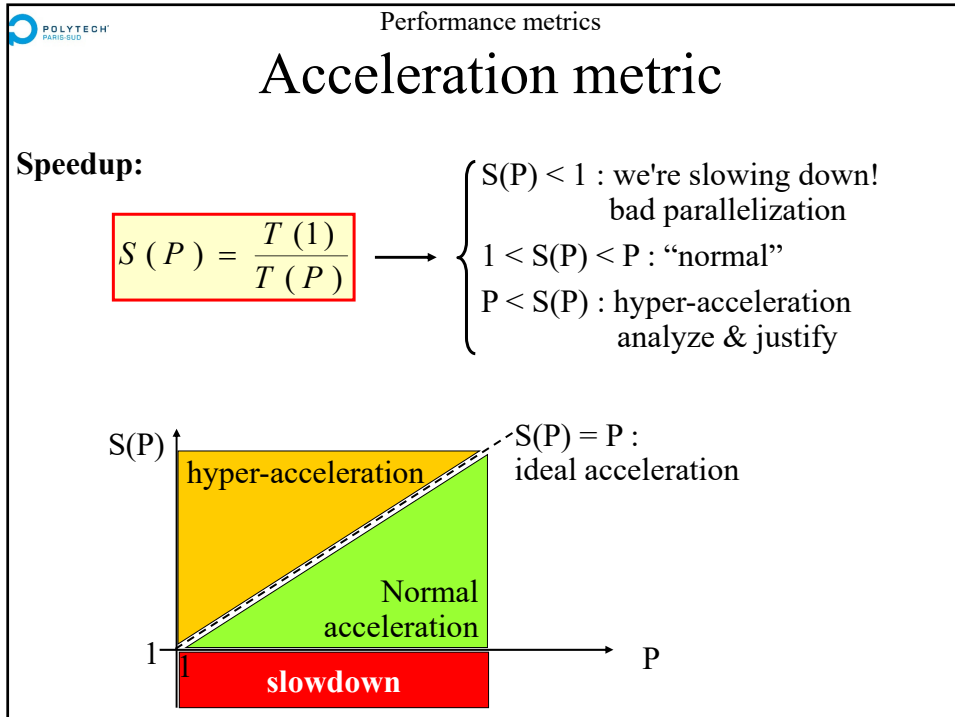
Execution time of a parallel/distributed application:



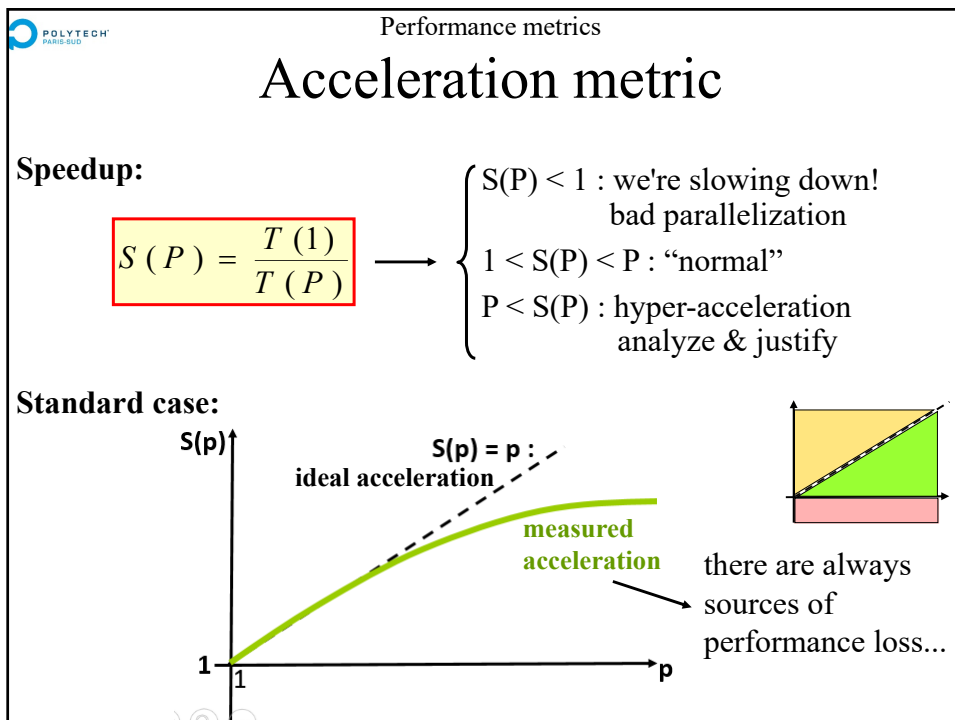
1. Is the execution time steadily decreasing?
 - As far as the number of resources exploited?
2. Is the deviation from the ideal curve acceptable?

Maintaining a constant decrease over a large number of resources, and close to the ideal decrease is very difficult.

10



11



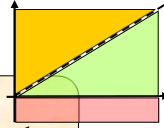
12

Acceleration metric

Cases of hyper-acceleration:

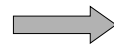
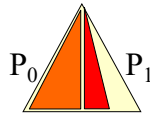
It's not magic, and it's not normal !

- The phenomenon must be analyzed and explained
- Correct an error or exploit an optimization



Examples of explanations :

- we no longer do the right operations (false result)
- the data fits in the total cache of the P processors
- we have modified the starting algorithm and converge faster (e.g. optimized genetic algorithm!).
- we look for a solution in a tree and stop the pgm



$$S(2) = 3$$

13

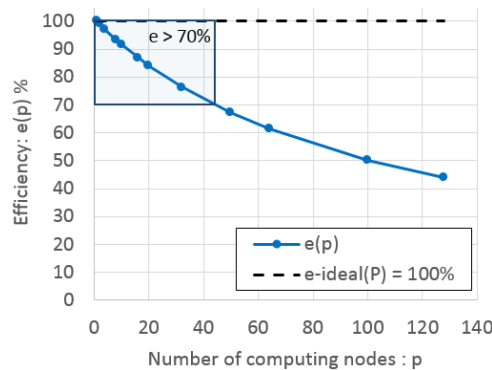
Efficiency metric

Efficacité :

$$e(P) = \frac{S(P)}{P}$$

Resource utilization rate, or fraction obtained of the ideal acceleration

- $e(P) \in [0;1]$, $\in [0\%;100\%]$
- $e(P) > 100\% \Leftrightarrow$ hyper-acceleration



The **user** is interested in the **acceleration** achieved

The **buyer** of the machine is interested in the **efficiency** of the applications

The **developer** is interested in **both**



14

Choice of the sequential reference

Which sequential execution to choose as a reference?

Same program running on a single processor?
Same algorithm implemented sequentially?
Best known sequential algorithm ?

Sequential compilation with the same compiler?
Compiling with the best sequential compiler ?

Sequential optimizations allowed by parallelization ?
Maximum sequential optimizations ?

Execution on a single processor of the parallel machine?
Execution on the best sequential machine ?

15

Choice of the sequential reference

Each sequential reference choice corresponds to:

- a different point of view
- a different business objective
- a different analysis objective

→ Make the choice corresponding to your problem

→ Clearly state this choice

Examples:

Final user → chooses HIS sequential pgm on HIS sequential machine

Parallel Code Developer → chooses HIS parallel pgm in ONE process of HIS parallel machine

16

3 – *Size Up* metrics

- Design of a code suitable for size up
- Size up metrics in execution time
- Size up metrics in time and resources
- Size up in 3 successive objectives
- Experimental examples

17

Size up metrics

Design of a code suitable for *size up*

1st objective: to be able to deal with bigger problems on more rsrcs

A code that **replicates** most of its data on all machines will always be limited by a machine's memory size...and will **not be suitable for size up**

RAM
Code

1 PC

3 PC

Successful speedup...

...but NO size up!

18

POLYTECH PARIS-SUD Size up metrics

Design of a code suitable for *size up*

1st objective: to be able to deal with bigger problems on more rsrcs

A code that **distributes** most of its data across all machines will be able to store more data on more machines...and **will be suitable for size up**

RAM Code 1 PC → 3 PC

Speedup successful & Size up compliant

First size up criterion passed

19

POLYTECH PARIS-SUD Size up metrics

Design of a code suitable for *size up*

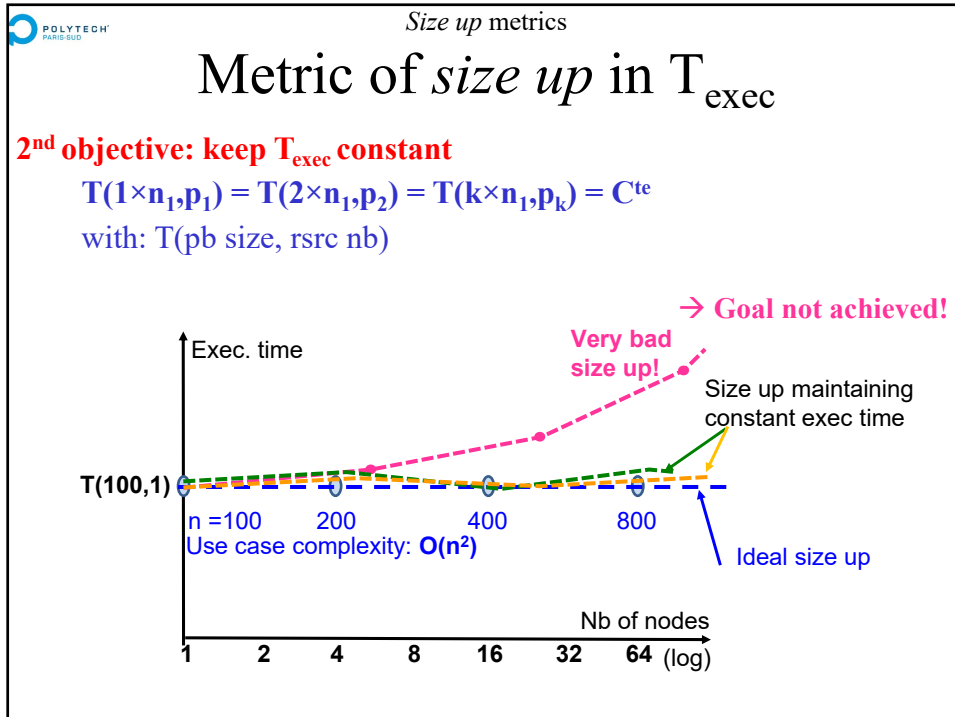
1st objective: to be able to deal with bigger problems on more rsrcs

→ Design of an initial distribution of data, and a minimum volume communication scheme

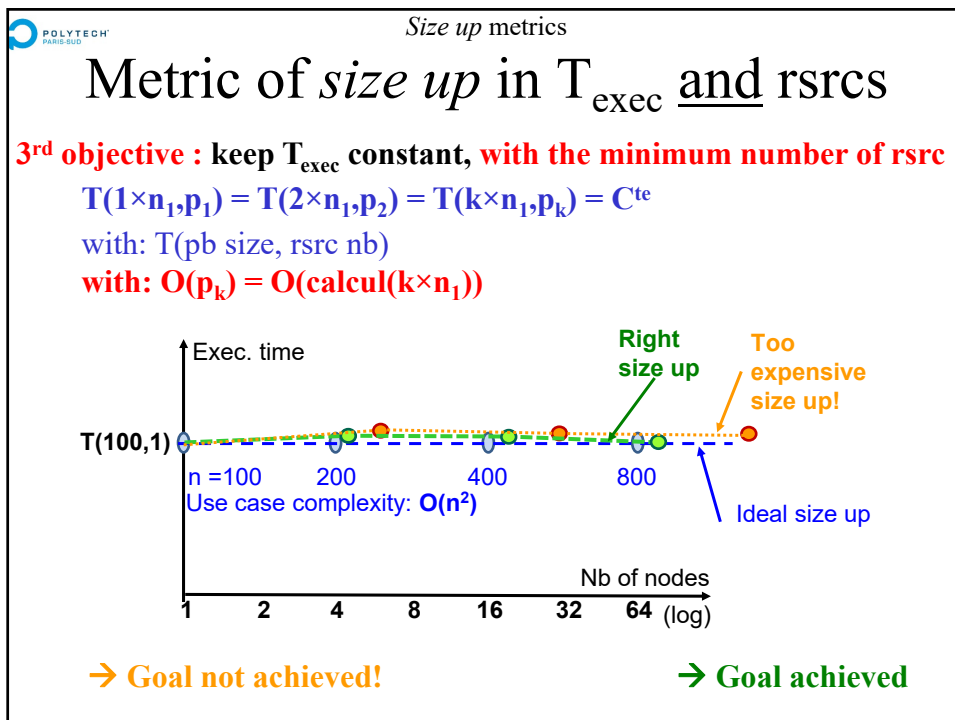
RAM Code 1 PC → 3 PC

- **Need to circulate initial data between nodes:** so that a node can continue calculations on data other than its own.
- **Need to circulate intermediate results between nodes:** so that one node can continue another node's calculations with its own data

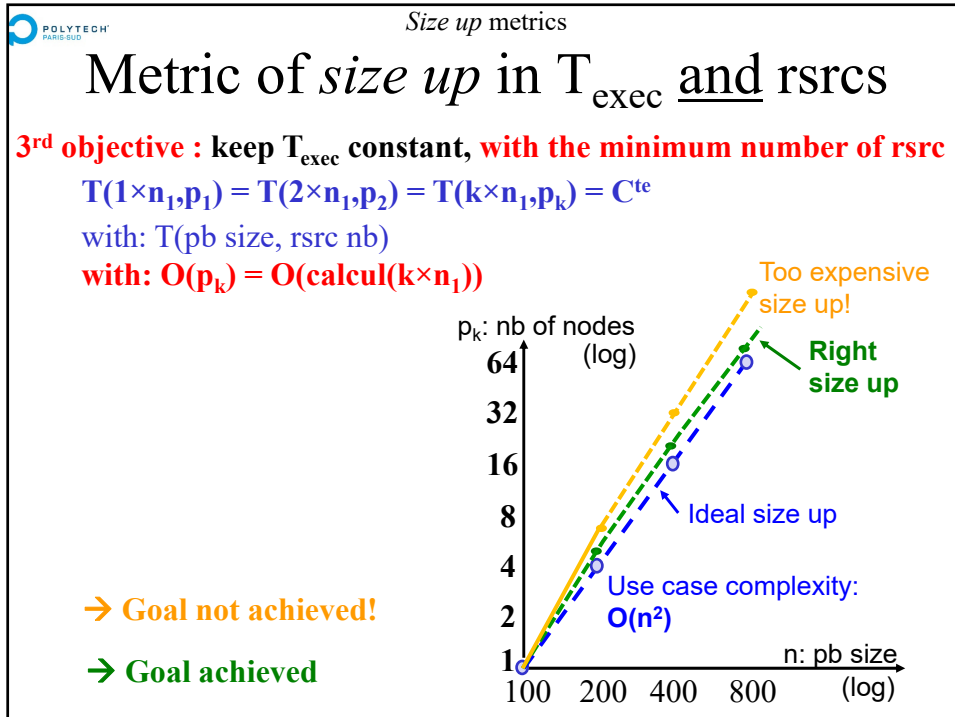
20



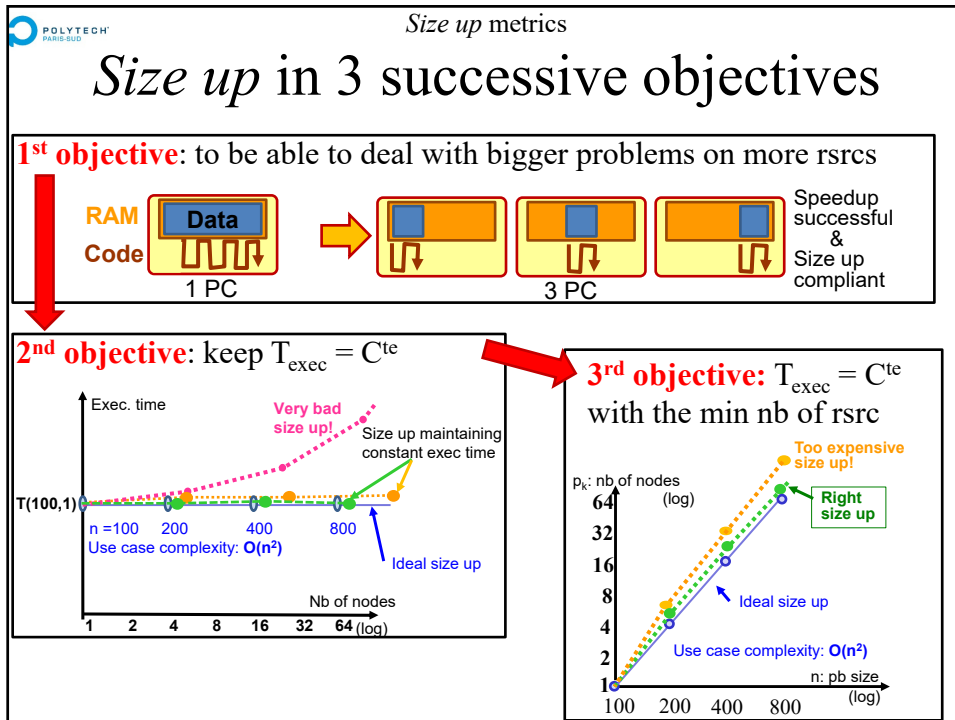
21



22





23



24

Size up metrics

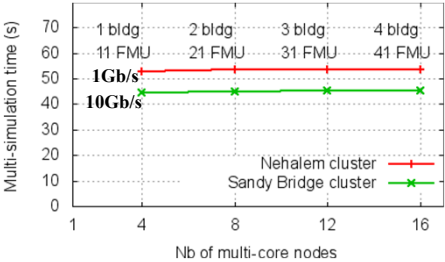
Experimental example

*Co-simulation of heat exchanges within buildings
weakly coupled (quasi-independent calculations)*

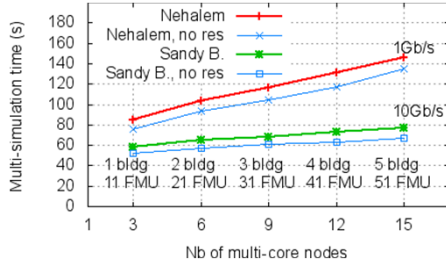
Objective: size up at constant time with minimum nb of rsrc

$T(1 \times 1 \text{ bldg}, p_1) = T(2 \times 1 \text{ bldg}, p_2) = T(k \times 1 \text{ bldg}, p_k) = C^{te}$

with: $O(p_k) = O(\text{calcul}(k \times 1 \text{ bldg})) \approx O(k)$



Nb of multi-core nodes




Nb of multi-core nodes

A lot of calculations and little comm. → Size up from 1Gb/s

Few calculations and little comm. → Size up from 10Gb/s

25



4 – Criteria for scaling

- A metric based on execution time
- *Size Up* + *Speedup* combined approach
- Scaling graphs

26

Criteria for scaling

A metric based on T_{exec}

As the size of the pb. grows, sequential exec. is no longer possible:

- not enough RAM, not enough disk...
- execution too long, resources cannot be monopolized...

- a sequential reference cannot be measured !
- we can't calculate Speedup !!

→ A *scaling* analysis should be built only on measures of T_{exec}

27

Criteria for scaling

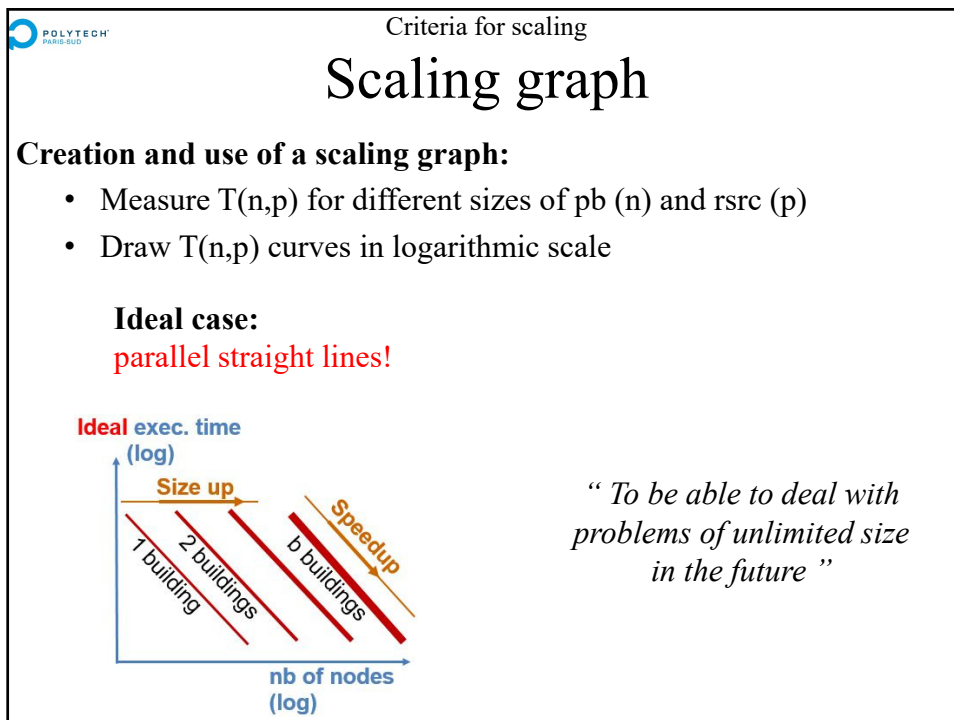
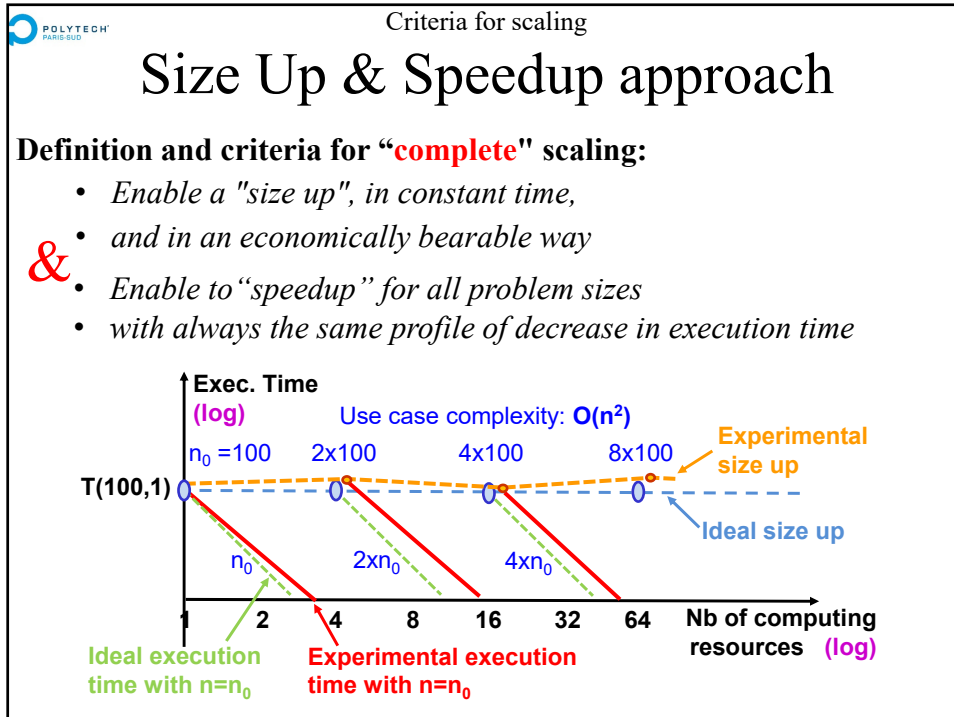
Size Up & Speedup approach

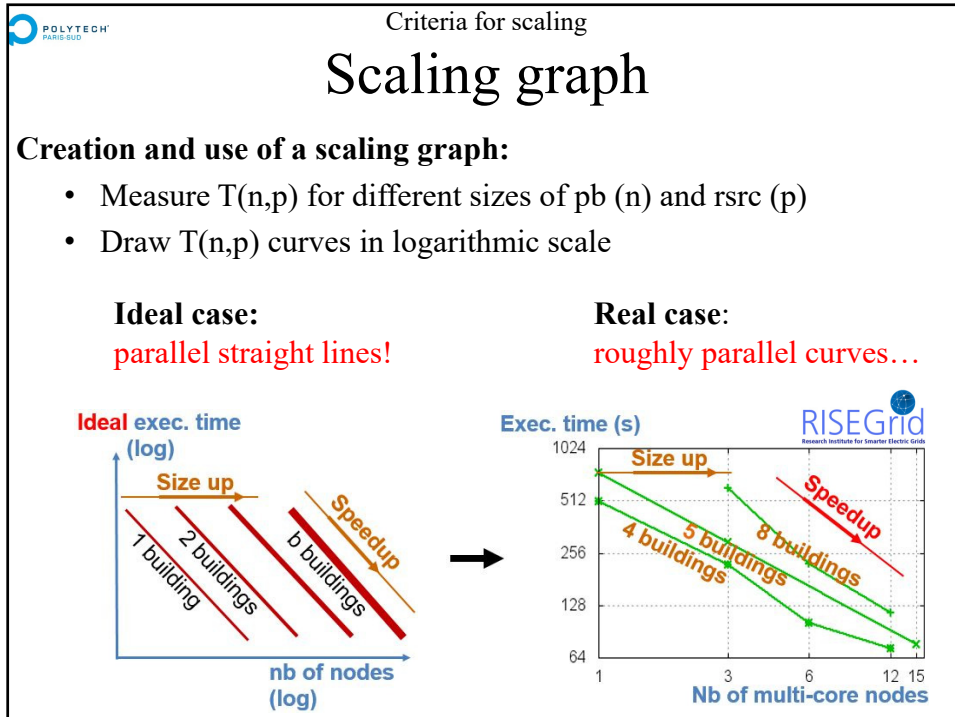
Definition and criteria for "simple" scaling:

- Enable a "size up", in constant time,
- and in an economically bearable way

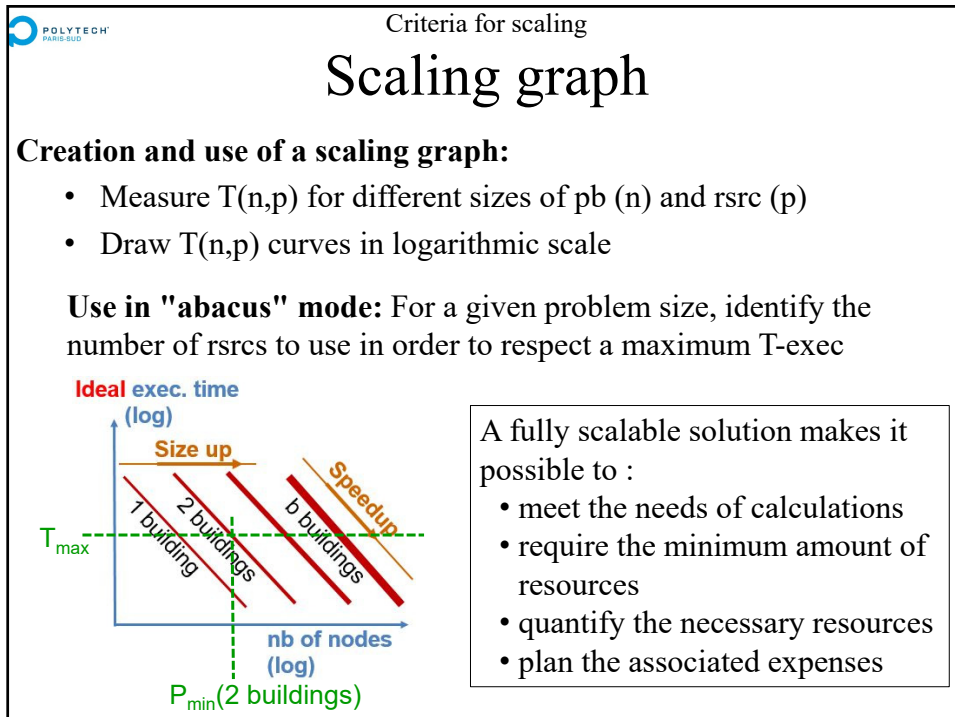
Level-3 size up

28





31

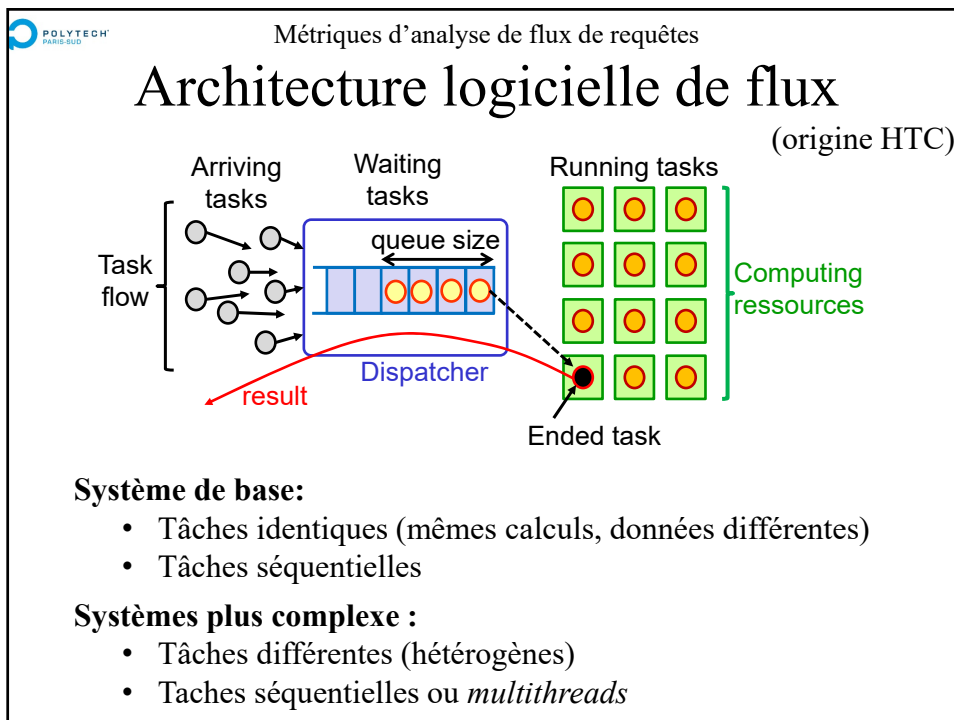


32

5 – Métriques d'analyse de flux de requêtes

- Architecture logicielle de flux
- Métrique de flux
- Analyse de pics de charge

33



34

POLYTECH PARIS-SUD

Métriques d'analyse de flux de requêtes

Métriques de flux

(origine HTC)

Métriques de flux :

1. Vitesse de traitement des tâches (tâches/s, tâches/min...)
2. Temps de traitement d'une tâche (T_{max} , T_{moy} ...)

→ Maintenir $T < T_{max}$, et $V > V_{flow}$

35

POLYTECH PARIS-SUD

Métriques d'analyse de flux de requêtes

Métriques de flux

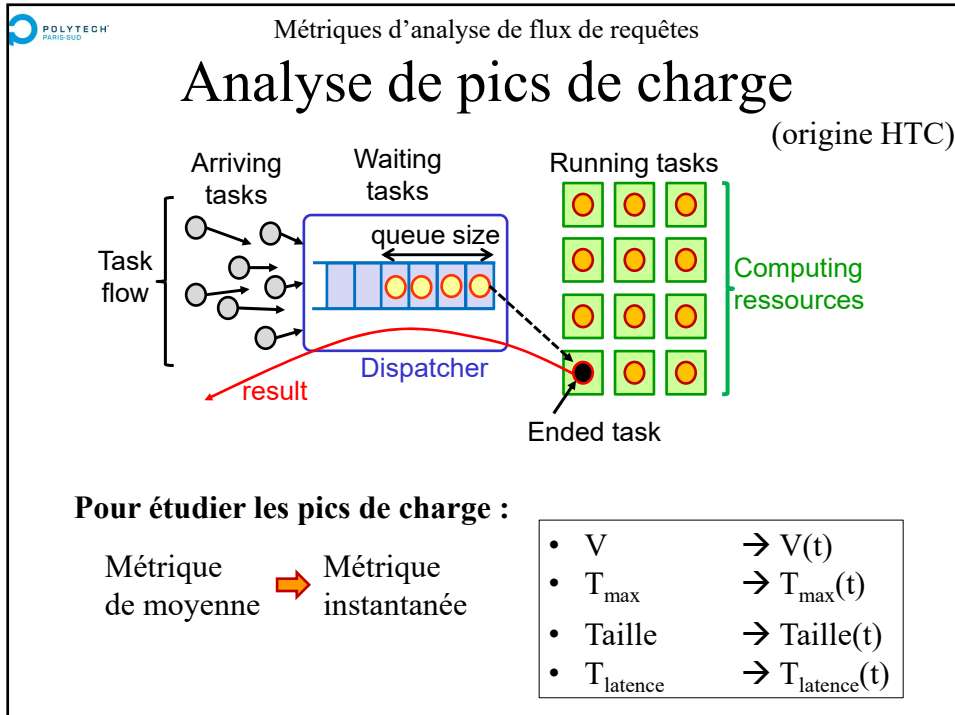
(origine HTC)

Métriques de flux :

3. Taille de la file d'attente (en nbr de tâches)
4. Temps de latence d'une tâche ($t_{démarrage} - t_{soumission}$)

→ Si *Taille de file* ou *Temps de latence* augmente
Alors augmenter le nombre de ressources de calcul
(ou améliorer le dispatcher...si pb)

36



37

POLYTECH PARIS-SUD

6 – Distributed execution of a Spark task graph

- Execution trace
- Performance measures & analysis

38

POLYTECH PARIS-SUD

Distributed execution of a Spark task graph

Execution trace

Spark job trace: on 10 Spark executors, with 3GB input file

```

DAGScheduler: Submitting 24 missing tasks from ShuffleMapStage 0 ...
TaskSchedulerImpl: Adding task set 0.0 with 24 tasks
...
TaskSetManager: Starting task 1.0 in stage 0.0 (TID 1, 172.20.10.14, executor 0, partition 1, ...)
TaskSetManager: Starting task 2.0 in stage 0.0 (TID 2, 172.20.10.11, executor 7, partition 2, ...)
...
TaskSetManager: Starting task 10.0 in stage 0.0 (TID 10, 172.20.10.11, executor 7, partition 10, ...)

TaskSetManager: Finished task 2.0 in stage 0.0 (TID 2) in 18274 ms ... (executor 7) (1/24)
TaskSetManager: Starting task 11.0 in stage 0.0 (TID 11, 172.20.10.7, executor 8, partition 11, ...)
TaskSetManager: Finished task 8.0 in stage 0.0 (TID 8) in 18459 ms ... (executor 8) (2/24)
...
TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks
                    have all completed, from pool
...

```

Annotations:

- Beggining of task graph execution** (points to DAGScheduler and TaskSchedulerImpl)
- Submitting the 10 first tasks on the 10 Spark executor processes** (points to TaskSetManager starting tasks)
- Submitting a new task when a previous one has finished** (points to TaskSetManager starting task 11.0)
- End of task graph execution** (points to TaskSchedulerImpl removing TaskSet 0.0)

39

POLYTECH PARIS-SUD

Distributed execution of a Spark task graph

Performance measures & analysis

Execution time as a function of the number of Spark executors


Ex. of Spark application run:

- from 1 up to 15 executors
- with 1 executor per node


Good overall decrease but plateaus appear !

Probable load balancing problem...


Ex: a graph of 4 parallel tasks



on 1
node: T

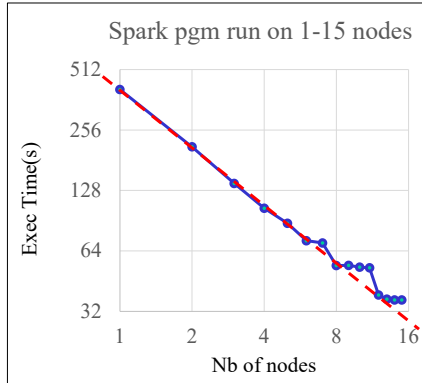


on 2
nodes: T/2



on 3
nodes: T/2

→ A plateau appears



Spark pgm run on 1-15 nodes

40

7 – Modélisations sur machines idéales

7.1 - Loi d'Amdahl

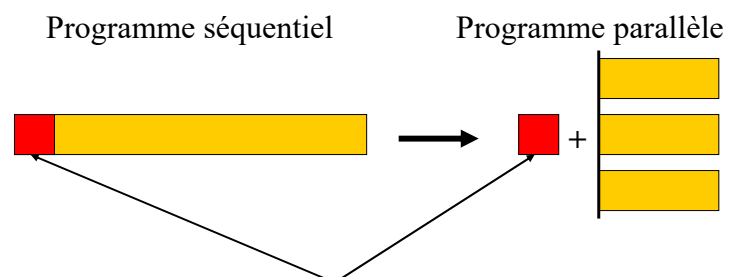
7.2 - Loi de Gustafson

7.3 - Lien Amdahl-Gustafson

41

Modélisations sur machines idéales

Loi d'Amdahl : Motivations



Fraction séquentielle du programme ...

... quel est son impact sur les performances ?

42

Loi d'Amdahl : Définitions

Hypothèses de départ :

$$T(1) = T_{seq} = T_s^a + T_p^a$$

↓ Temps de la partie *parallélisable* du programme (au sens de Amdahl)
 ↓ Temps de la partie *séquentielle* du programme (au sens de Amdahl)

$$T(P) = T_s^a + T_p^a / P + \text{Overhead}$$

$$T(P) = T_s^a + T_p^a / P$$

Hyp : machine parallèle idéale

- Synchro sans surcoût !
- Msgs instantannés !
- ...

43

Loi d'Amdahl : Définitions

Définition de la fraction séquentielle au sens de Amdahl :

$$f_s^a = \frac{T_s^a}{T(1)} = \frac{T_s^a}{T_s^a + T_p^a} \in [0;1]$$

Et la fraction parallèle:

$$f_p^a = \frac{T_p^a}{T(1)} = \frac{T_p^a}{T_s^a + T_p^a} \in [0;1]$$

D'où :

$$f_s^a + f_p^a = 1$$

Réécriture du temps d'exécution:

$$T(P) = T_s^a + T_p^a / P$$

$$T(P) = f_s^a \cdot T(1) + (1 - f_s^a) \cdot T(1) / P$$

44

Loi d'Amdahl : Définitions

Conséquence sur le *speedup* :

Par définition du *Speedup* : $S^a(P) = \frac{T(1)}{T(P)}$

Donc : $S^a(P) = \frac{T(1)}{f_s^a \cdot T(1) + (1 - f_s^a) \cdot T(1) / P}$

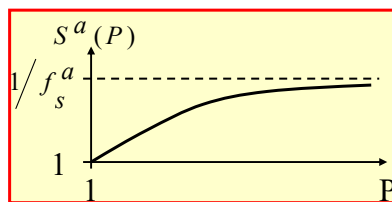
Soit : $S^a(P) = \frac{1}{f_s^a + \frac{1 - f_s^a}{P}}$ $\lim_{P \rightarrow \infty} S^a(P) = \frac{1}{f_s^a} \quad S^a(P) \leq \frac{1}{f_s^a}$

$S^a(P) = P \cdot \frac{1}{1 + f_s^a \cdot (P - 1)}$ $S^a(P) = S^{ideal}(P) \cdot \frac{1}{1 + f_s^a \cdot (P - 1)}$

45

Loi d'Amdahl : Impact sur le *speedup*

Allure du *speedup* :



Exemple (a.n.) :

$$f_s^a = 1\% \Rightarrow \begin{cases} S^a(P) < 100 \\ S^a(100) = 50.3 \end{cases}$$

Petite fraction séquentielle →
grosse limitation pour P grand!

46

Loi d'Amdahl : Confrontation à la réalité

En réalité les overhead ne sont pas négligeables :

$$T(P) = T_s^a + T_p^a / P + \text{Overhead}$$

Temps de communication
Temps de synchronisation
Temps d'ordonnancement
...

$$\Rightarrow S^{\text{réel}}(P) < S^a(P)$$

Pourtant on n'observe pas de si mauvais résultats (en général) !

“On sait obtenir de bon *speedup* sur de grosses machines parallèles”

→ { Les fractions séquentielles sont généralement faibles ?
Le modèle ne correspond pas toujours à la réalité ?

→ Loi de Gustafson

47

7 – Modélisations sur machines idéales

7.1 - Loi d'Amdahl

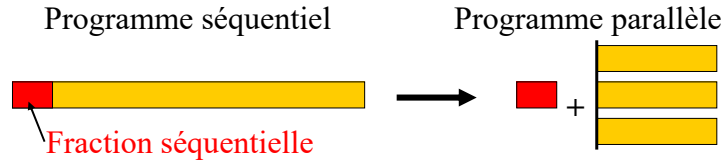
7.2 - Loi de Gustafson

7.3 - Lien Amdahl-Gustafson

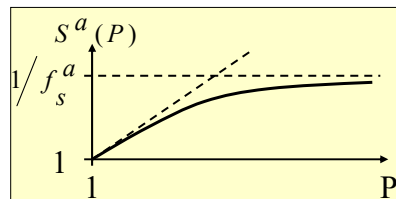
48

Loi de Gustafson : Motivation

1 – Impact de la fraction séquentielle (selon Amdahl) ?



2 – Mais on observe qu’Amdahl n’est pas réaliste (trop pessimiste) :



Construire un modèle plus proche de la réalité ...

49

Loi de Gustafson : Définitions

Idée/Constataion de base :

En pratique : on ne veut pas diminuer le temps d'exécution T_0 ,
mais augmenter la quantité de travail W traitée en T_0



- On traite W_0 en séquentiel en : $T(1, W_0)$
- On traite W en parallèle en : $T(P, W)$

Et on choisit P et W tels que : $T(P, W) = T(1, W_0) = T_0$

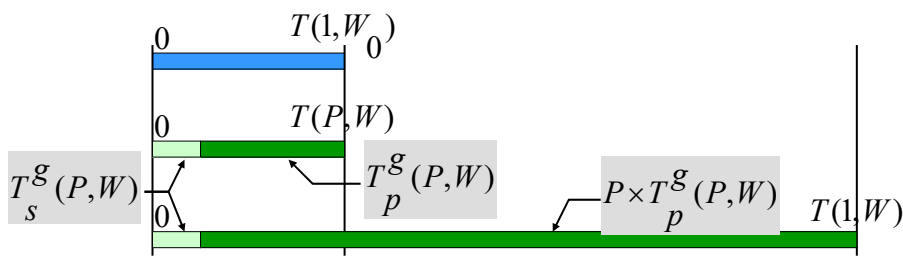
50

Loi de Gustafson : Définitions

Temps que *durerait* le traitement séquentiel de W :

$$T(1, W) = T_s^g(P, W) + P \times T_p^g(P, W) - \text{Overhead}$$

Temps de la partie *parallélisée* sur P processeurs (au sens de Gustafson)
 Temps de la partie séquentielle du programme (au sens de Gustafson)



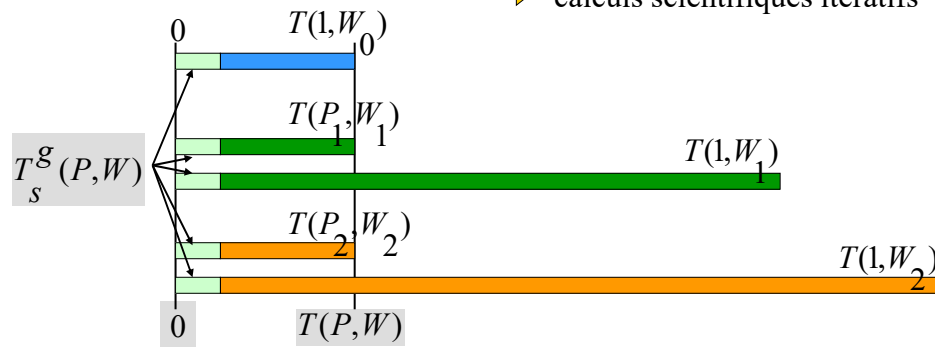
51

Loi de Gustafson : Définitions

Hypothèse : partie séquentielle constante

$$T_s^g(P_1, W_1) = T_s^g(P_2, W_2) = T_s^g(P, W) = Cte$$

Vrai notamment pour des calculs scientifiques itératifs



52

Loi de Gustafson : Définitions

Introduction de la fraction séquentielle de Gustafson :

$$f_s^g / T_s^g = f_s^g \times T(P, W), \quad f_s^g \in [0, 1]$$

$$\left. \begin{array}{l} T(P, W) = Cte, \text{ par définition} \\ \text{avec : } P = P_g(W) \\ T_s^g = Cte, \text{ par hypothèse} \end{array} \right\} \rightarrow f_s^g = Cte$$

Nouvelle expression du temps séquentiel (estimé) :

$$T(1, W) = T_s^g(P, W) + P \times \frac{T_s^g(P, W)}{P}$$

$$\downarrow$$

$$T(1, W) = f_s^g \times T(P, W) + P \times (1 - f_s^g) \times T(P, W)$$

53

Loi de Gustafson : Définitions

Speedup fonction de la fraction séquentielle :

$$S(P, W) = \frac{T(1, W)}{T(P, W)}$$

$$S^g(P, W) = \frac{f_s^g \cdot T(P, W) + P \cdot (1 - f_s^g) \cdot T(P, W)}{T(P, W)}$$

$$S^g(P, W) = f_s^g + P \cdot (1 - f_s^g)$$

$$S^g(P, W) = f_s^g + P \cdot (1 - f_s^g) \quad \text{Speedup au sens de Gustafson}$$

Mais la modélisation de Gustafson considère uniquement des couples (P_i, W_i) tels que $T(P_i, W_i) = T(1, W_0)$

$$\hookrightarrow S^g(P(W)) = f_s^g + P(W) \cdot (1 - f_s^g)$$

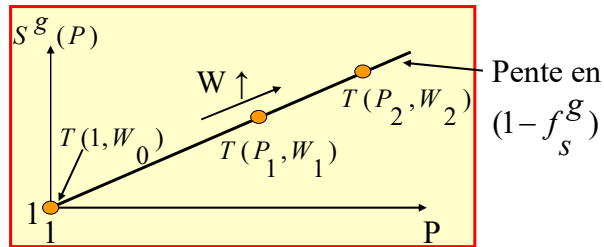
54

Loi de Gustafson : Impact sur le *speedup*

Speedup non borné :

$$\lim_{P \rightarrow \infty} S^g(P) = \infty$$

Allure du *speedup* :



Bilan de la loi de Gustafson :

- Adopte le **point de vue utilisateur** : augmenter W à T -exec constant
- **Hypothèse optimiste** de partie séquentielle constante ($T_s^g(P, W) = Cte$)

55

7 – Modélisations sur machines idéales

7.1 - Loi d'Amdahl

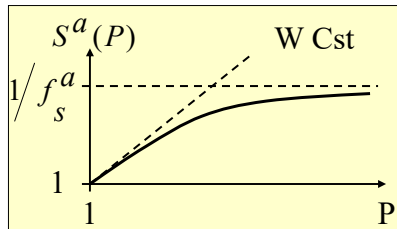
7.2 - Loi de Gustafson

7.3 - Lien Amdahl-Gustafson

56

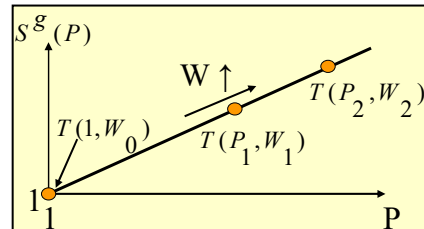
Amdahl – Gustafson : Comparaison

Amdahl et Gustafson aboutissent à des conclusions différentes :



Amdahl

Etudie **l'extensibilité forte** (*strong scalability*) : capacité à décroître T-exec qd seule la taille de la machine croit.



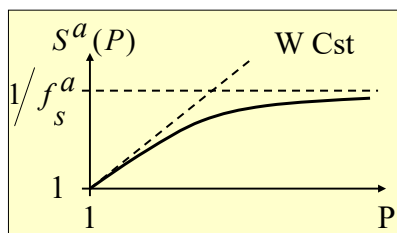
Gustafson

Etudie **l'extensibilité faible** (*weak scalability*) : capacité à maintenir T-exec qd les tailles de la machine et du pb croissent

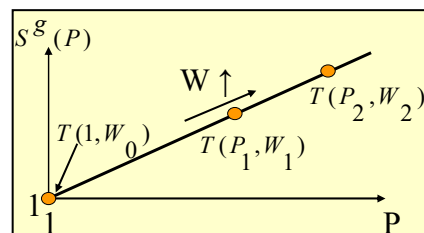
57

Amdahl – Gustafson : Comparaison

Amdahl et Gustafson aboutissent à des conclusions différentes :



Amdahl



Gustafson

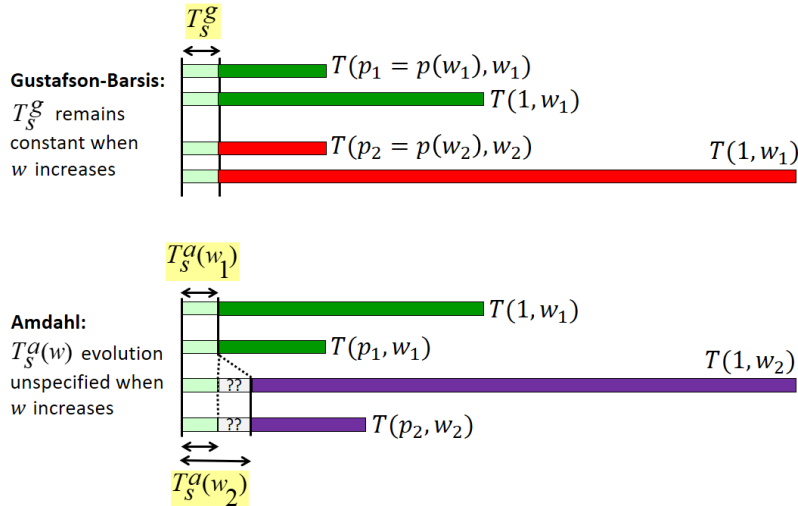
En fait Amdahl et Gustafson :

- Font des hypothèses différentes
- Définissent des fractions séquentielles différentes
- **Ne sont pas incompatibles**

58

Amdahl – Gustafson : Comparaison

Différence de modélisation de la fraction séquentielle :



59

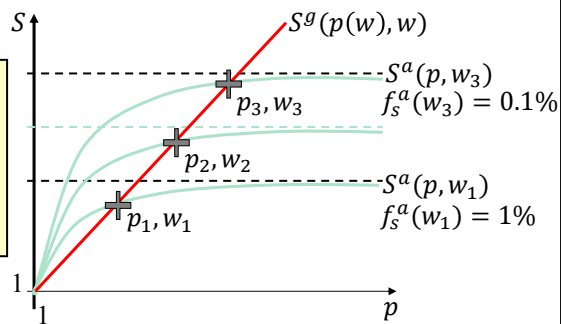
Amdahl – Gustafson : Relation entre SU

Expressions complètes des speedup

$$S^a(P, w) = \frac{1}{f_s^a(w) + \frac{1 - f_s^a(w)}{P}}$$

$$S^g(p_g(w)) = f_s^g + p_g(w) \cdot (1 - f_s^g)$$

Allures des speedup



En augmentant P et W (selon Gustafson) à f_s^g constant :

→ $S^g(P)$ intersecte différent $S^a(P)$, avec des $f_s^a(w)$ décroissant

60

Amdahl – Gustafson : Bilan

Amdahl : W Cte, $P \uparrow$, T -exec \downarrow

Réaliste quand on travaille à W Cte

Réaliste quand on augmente P pour diminuer T -exec

Informaticien

Gustafson : $W \uparrow$, $P \uparrow$, T -exec Cte

Réaliste quand on augmente W sur *certain*s pbs

Ex : plus de cycles de calculs lors de simulations

Utilisateur

En général :

Valeurs des courbes réelles moins bonnes à cause des *overhead*

Allures des courbes réelles entre Amdahl et Gustafson

61

QUIZ

62

QUIZ

1. Vous devez acheter une application distribuée « A » pour la faire tourner sur 4 noeuds, et vous disposez de l'information suivante :
 - « l'application A1 exhibe un speedup de 3.9 sur 4 noeuds »
 - « l'application A2 exhibe un speedup de 3.0 sur 4 noeuds »

Que faites-vous ?

2. Vous devez acheter une application distribuée « A », et vous disposez de l'information suivante :
 - « l'application A1 traite $2 \cdot 10^4$ documents/s sur 10 noeuds »
 - « l'application A2 traite $3 \cdot 10^4$ documents/s sur 10 noeuds »

Que faites-vous si vous avez 10 noeuds ?

Que faites-vous si vous avez 20 noeuds ?

63

QUIZ

3. Citez des métriques de performances absolues et de performances relatives
4. Qu'est ce qui peut empêcher un système (hard & soft) de passer à l'échelle ?
5. On considère une application sous la forme d'un graphe de tâches très hétérogènes (des traitements longs et d'autres courts)
 - Si on augmente le nombre de noeuds sans augmenter le nombre de tâches (i.e: la taille du pb), va-t-on obtenir un bon speedup ?
 - Un « passage à l'échelle » reste-t-il possible ?

64

QUIZ

6. Vous devez augmenter le nombre de nœuds de calcul pour traiter des problèmes plus importants dans le même temps.

Au lieu de cela on vous propose de passer tous vos nœuds de 1 à 2 processeurs, sans autre changement dans votre cluster (solution moins onéreuse...).

Que devez-vous vérifier avant d'opter pour un passage à l'échelle avec cette solution ?

65

Appendix

A - Methodology for measurement of execution time

66

POLYTECH PARIS-SUD

Mesure des temps d'exécution

Méthodologie de mesures


Mesures externes :

```
>time myAppli
>/usr/bin/time myAppli
>times myAppli
>timex myAppli
.....
```


12.002u	user
0.128s	system
12.150	total

Nom et fonctionnement variables selon le système utilisé !!

Fréquemment : total > user + system !!



Simple à utiliser
Pas de modifications des codes sources



Peu précis: ± 0.5s

67

POLYTECH PARIS-SUD

Mesure des temps d'exécution

Méthodologie de mesures


Mesures internes :

```
time ()
clock ()
gettimeofday ()
omp_get_wtime ()
```


→ Compte les clicks d'horloge

→ Compte le temps écoulé en s

- Toutes ces routines ne sont pas toujours disponibles !
- "gettimeofday" est en général une bonne solution.
- Parfois il existe des outils plus précis pour mesurer de petites durées.



Plus précis que les mesures externes

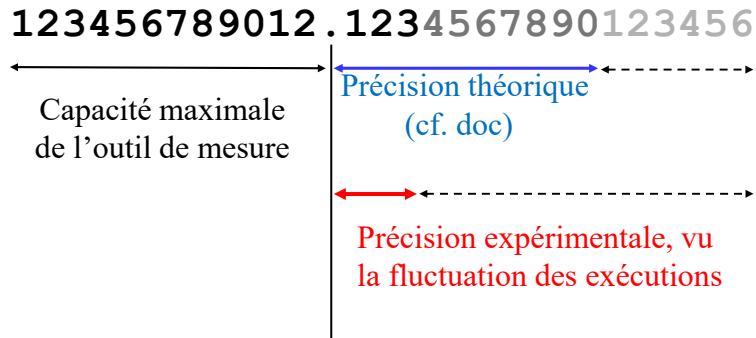


Besoin de modifier le code source
Pas toujours totalement portable

68

Méthodologie de mesures

Précision des outils et des mesures :



- Ne pas tenir compte de trop de décimales!
- Faire attention à ne pas déborder la capacité de mesure!

69

Méthodologie de mesures

Problème fréquent :

Test en mode exclusif (mono-user).
Outil de mesure à 1ms de précision.



Fluctuation de 500ms
d'une exécution à l'autre !!

Et plus encore avec la
montée en fréquence
automatique des procs.
(effet de "chauffe")

Démarche conseillée :

- Mesurer les fluctuations, ne pas les ignorer
(le *warm up* des processeurs peut perturber les premières)
- Ne pas donner que les valeurs moyennes
- Mesurer des temps > 10s si possible

70

Méthodologie de mesures

Stocker des meta-données sur les conditions de mesure :

- **Date de l'exécution**
- **Auteur(s) du test**
- Outil(s) de mesure utilisé(s)
- Caractéristiques de la machine :
RAM, Cache, Processeurs, ...
- OS utilisé (nom et version)
- Compilateur utilisé (nom et version)
- Options de compilation utilisées
- Test en multi-user/mono-user ?
- Présence d'IO dans le test ?
- Configuration du programme de test : taille des données, ...

On oublie souvent
(et rapidement) à
quel benchmark se
réfère une série de
mesures !

On manque souvent
de détail sur les
conditions de
réalisation d'une
série de mesures !

71

Performance, efficiency and scalability metrics



72