

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Green Computing - CM 5

Techniques de conception pour des logiciels plus économes

Slides réalisés avec l'aide Romain Lefeuvre, doctorant @DiverSE

Quentin Perez
quentin.perez@insa-rennes.fr
4INFO
2025-2026

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Le contexte d'exécution / mesure

Les travaux présentés dans ces slides et les leviers techniques permettant d'économiser de l'énergie au niveau logiciel sont toujours à considérer dans leurs contextes !

Une étude sur un objet (programme, un service, etc.) dans un contexte A pourra avoir des résultats différents dans un contexte B.

La **consommation d'énergie** est **fortement impactée par le contexte et l'environnement matériel et logiciel d'exécution** !

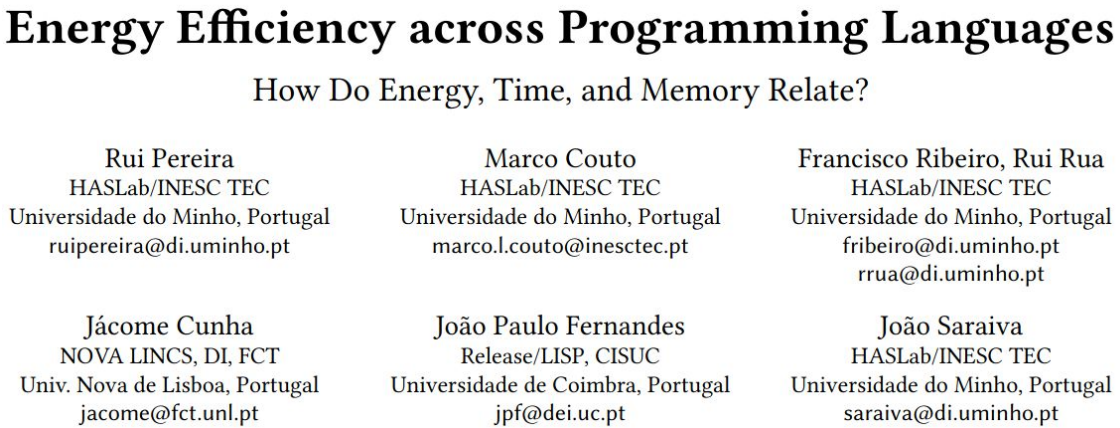
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Contexte d’exécution / de mesure et impact sur les résultats de consommation

Illustration de l'influence du contexte : le langage de programmation et la consommation

Pour illustrer le caractère contextuel, nous allons étudier une publication "connue" qui fait l'objet de nombreux débats...

https://sites.google.com/view/energy-efficiency-languages



Energy Efficiency across Programming Languages

How Do Energy, Time, and Memory Relate?

Rui Pereira
HASLab/INESC TEC
Universidade do Minho, Portugal
ruipereira@di.uminho.pt

Marco Couto
HASLab/INESC TEC
Universidade do Minho, Portugal
marco.l.couto@inesctec.pt

Francisco Ribeiro, Rui Rua
HASLab/INESC TEC
Universidade do Minho, Portugal
fribeiro@di.uminho.pt
rrua@di.uminho.pt

Jácome Cunha
NOVA LINCS, DI, FCT
Univ. Nova de Lisboa, Portugal
jacome@fct.unl.pt

João Paulo Fernandes
Release/LISP, CISUC
Universidade de Coimbra, Portugal
jpf@dei.uc.pt

João Saraiva
HASLab/INESC TEC
Universidade do Minho, Portugal
saraiva@di.uminho.pt

[Pereira17]

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Illustration de l'influence du contexte : le langage de programmation et la consommation

Voici l'avis des "Jean-Michel informaticiens" sur LinkedIn... 😭 😂

Chris Lefevre • 3e et + — + Suivre
CTO & Digital Specialist at celestetic. *Je partage des actus IA ...*
6 mois •

Coder en Python fait de vous un (gros) pollueur

Choquant, n'est-ce pas ? Pourtant, c'est la réalité que révèlent les dernières études sur l'efficacité énergétique des langages de programmation.

Saviez-vous que Python peut consommer jusqu'à 45 fois plus d'énergie que C pour des tâches équivalentes ? C'est comme si vous laissiez tourner votre climatiseur 24/7 au lieu de l'allumer seulement quand nécessaire.

Mais attendez, il y a plus :
- JavaScript s'en sort plutôt bien côté consommation
- Surprise : même PHP fait 3 fois mieux que Python en termes d'efficacité énergétique !

Voici ce qu'il faut retenir :

1. Les langages compilés comme C, Rust et C++ sont les champions de l'efficacité énergétique.

2. Les langages interprétés (Python en tête, suivi de Ruby et Perl) sont les plus gourmands en énergie.

3. Le choix du langage dépend de nombreux facteurs : type de projet, rapidité de développement, maintenabilité...

Chris Lefevre Auteur — 6 mois
CTO & Digital Specialist at celestetic. *Je partage des actus IA et ...*

Joel parfait Kuate en vrai, j'ai jamais compris et engouement pour Python, je trouve la syntaxe moche en plus 🤔

J'aime | Répondre

philippe joulin • 3e et + — 6 mois
LeadDev WEB, Dev FullStack, Agitateur de conscience

Le python ressemble beaucoup au php/javascript, si tu connais ces 2 langages tu comprends python tout de suite :)
Après python est bas niveau contrairement à PHP et Javascript, tout comme nodeJs, disons que c'est une question de préférence :D

J'aime | Répondre

philippe joulin • 3e et + — 6 mois
LeadDev WEB, Dev FullStack, Agitateur de conscience

Joel parfait Kuate En effet ce n'est pas le langage qui est la cause, plutôt son serveur d'hébergement pour le web ou la machine qui va fonctionner pour les cas de logiciels.
Mais l'air de rien le plus gros consommateur reste Internet, les échanges permanent de données à travers le monde consc ... plus

J'aime · 1 | Répondre

Joel parfait Kuate • 3e et + — 6 mois
Entrepreneur Tech, auteur, consultant en transformation nu...

Effectivement, Python se distingue par son orientation vers la simplicité et la lisibilité du code, ainsi que par sa polyvalence dans divers domaines tels que le développement web, l'analyse de données et l'intelligence artificielle. En termes de niveau, Python est en effet plus proche du langage machine que PHP et ... plus

J'aime | Répondre

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Illustration de l'influence du contexte : le langage de programmation et la consommation - Questions de recherche

RQ1 : "Can we compare the energy efficiency of software languages?"

RQ2 : "Is the faster language always the most energy efficient?"

"~~How does memory usage relate to energy consumption?~~"

RQ3 : "Can we automatically decide what is the best programming language considering energy, time, and memory usage?"

Question subsidiaire : "Le comparatif des langages présenté ici est-il représentatif d'une utilisation "classique industrielle"" ?

[Pereira17]

Illustration de l'influence du contexte : le langage de programmation et la consommation - Questions de recherche

Comparer des langages bien que semblant facile (#yakafocon) n'est pas quelque chose de simple :

- **Variabilité ! :**
  - **Langages compilés VS interprétés**
  - **Utilisation de machine virtuelle**
  - **Optimisations du compilateur pour la création de l'exécutable**
  - **Présence ou non d'un garbage collector**
  - **Utilisation de librairies tierces**
  - **…**

[Pereira17]

Illustration de l'influence du contexte : le langage de programmation et la consommation - Méthodologie

Pereira *et al.* ont utilisé un benchmark "connu" permettant de comparer les performances en termes de rapidité des langages :

The Computer Langage Benchmark Game

https://benchmarksgame-team.pages.debian.net/benchmarksgame/index.html

⇒ **Positif pour faire des comparaisons temps VS énergie**

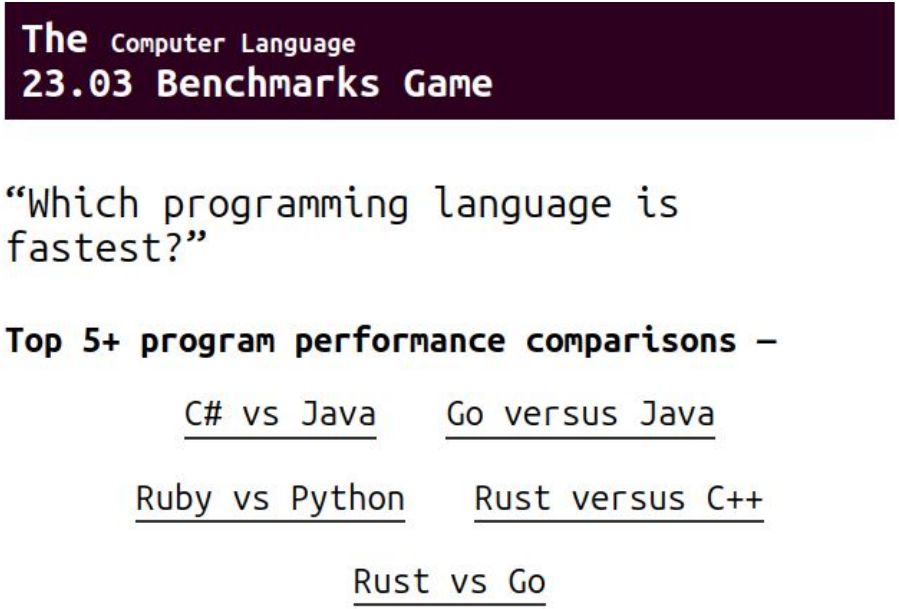


Illustration de l'influence du contexte : le langage de programmation et la consommation - Méthodologie

Pereira *et al.* ont utilisé un benchmark "connu" permettant de comparer les performances en termes de rapidité des langages :

The Computer Langage Benchmark Game

https://benchmarksgame-team.pages.debian.net/benchmarksgame/index.html

⇒ Positif pour faire des comparaisons temps VS énergie

RQ1 ⇒ oui, on peut comparer l'efficacité énergétique des langages, MAIS... car il y a un mais

Benchmark	Description	Input
n-body	Double precision N-body simulation	50M
fannkuch-redux	Indexed access to tiny integer sequence	12
spectral-norm	Eigenvalue using the power method	5,500
mandelbrot	Generate Mandelbrot set portable bitmap file	16,000
pidigits	Streaming arbitrary precision arithmetic	10,000
regex-redux	Match DNA 8mers and substitute magic patterns	fasta output
fasta	Generate and write random DNA sequences	25M
k-nucleotide	Hashtable update and k-nucleotide strings	fasta output
reverse-complement	Read DNA sequences, write their reverse-complement	fasta output
binary-trees	Allocate, traverse and deallocate many binary trees	21
chameneos-redux	Symmetrical thread rendezvous requests	6M
meteor-contest	Search for solutions to shape packing puzzle	2,098
thread-ring	Switch from thread to thread passing one token	50M

Illustration de l'influence du contexte : le langage de programmation et la consommation - Méthodologie

The Computer Langage Benchmark Game

⇒ Positif pour faire des comparaisons temps VS **énergie**, **MAIS... car il y a un mais...**

Cale pose la question de la représentativité du contexte "réel" d'utilisation de ces langages ⇒ Génère-t-on souvent des ensembles de Mandelbrot...?

Benchmark	Description	Input
n-body	Double precision N-body simulation	50M
fannkuch-redux	Indexed access to tiny integer sequence	12
spectral-norm	Eigenvalue using the power method	5,500
mandelbrot	Generate Mandelbrot set portable bitmap file	16,000
pidigits	Streaming arbitrary precision arithmetic	10,000
regex-redux	Match DNA 8mers and substitute magic patterns	fasta output
fasta	Generate and write random DNA sequences	25M
k-nucleotide	Hashtable update and k-nucleotide strings	fasta output
reverse-complement	Read DNA sequences, write their reverse-complement	fasta output
binary-trees	Allocate, traverse and deallocate many binary trees	21
chameneos-redux	Symmetrical thread rendezvous requests	6M
meteor-contest	Search for solutions to shape packing puzzle	2,098
thread-ring	Switch from thread to thread passing one token	50M

Illustration de l'influence du contexte : le langage de programmation et la consommation - Méthodologie

27 langages comparés avec différents paradigmes :

- fonctionnels
- impératifs
- orientés objets
- de scripts

Paradigm	Languages
Functional	Erlang, F#, Haskell, Lisp, Ocaml, Perl, Racket, Ruby, Rust;
Imperative	Ada, C, C++, F#, Fortran, Go, Ocaml, Pascal, Rust;
Object-Oriented	Ada, C++, C#, Chapel, Dart , F#, Java, JavaScript, Ocaml, Perl, PHP, Python, Racket, Rust, Smalltalk, Swift, TypeScript;
Scripting	Dart, Hack, JavaScript, JRuby, Lua, Perl, PHP, Python, Ruby, TypeScript;

Illustration de l'influence du contexte : le langage de programmation et la consommation - Méthodologie

Mesure de consommation :

- RAPL en utilisant une bibliothèque C
- Chaque programme est exécuté et mesuré 10x

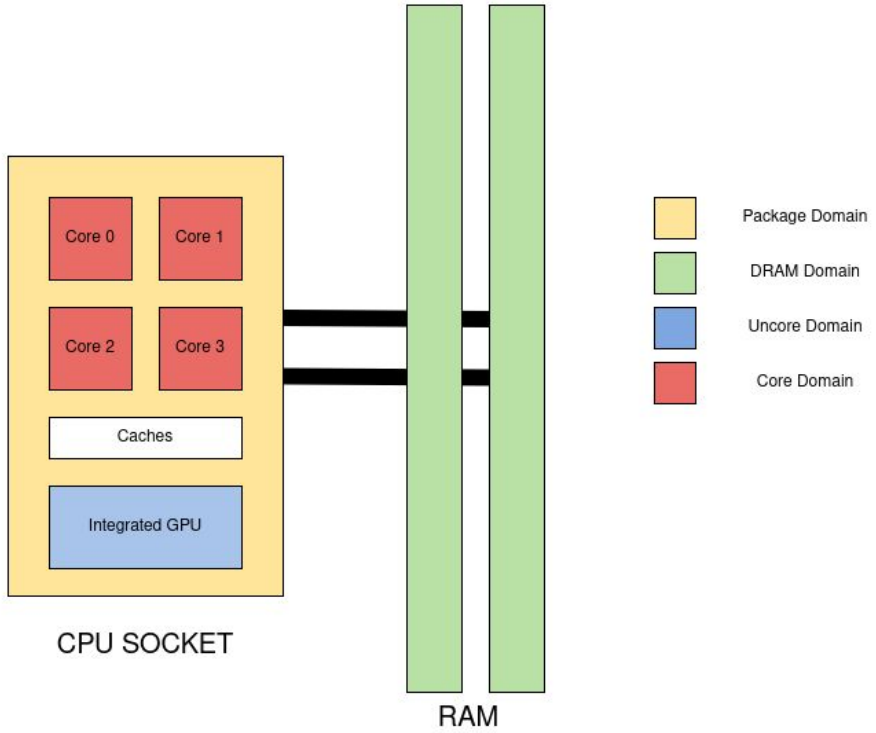


Illustration de l'influence du contexte : le langage de programmation et la consommation - Résultats

C est selon le benchmark le langage le plus économe et rapide parmi les langages compilés dans le cas de **binary-trees**

	binary-trees			
	Energy	Time	Ratio	Mb
(c) C	39.80	1125	0.035	131
(c) C++	41.23	1129	0.037	132
(c) Rust $\Downarrow_2$	49.07	1263	0.039	180
(c) Fortran $\Uparrow_1$	69.82	2112	0.033	133
(c) Ada $\Downarrow_1$	95.02	2822	0.034	197
(c) Ocaml $\downarrow_1 \Uparrow_2$	100.74	3525	0.029	148
(v) Java $\uparrow_1 \Downarrow_{16}$	111.84	3306	0.034	1120
(v) Lisp $\downarrow_3 \Downarrow_3$	149.55	10570	0.014	373
(v) Racket $\downarrow_4 \Downarrow_6$	155.81	11261	0.014	467
(i) Hack $\uparrow_2 \Downarrow_9$	156.71	4497	0.035	502
(v) C# $\downarrow_1 \Downarrow_1$	189.74	10797	0.018	427
(v) F# $\downarrow_3 \Downarrow_1$	207.13	15637	0.013	432
(c) Pascal $\downarrow_3 \Uparrow_5$	214.64	16079	0.013	256
(c) Chapel $\uparrow_5 \Uparrow_4$	237.29	7265	0.033	335
(v) Erlang $\uparrow_5 \Uparrow_1$	266.14	7327	0.036	433
(c) Haskell $\uparrow_2 \Downarrow_2$	270.15	11582	0.023	494
(i) Dart $\downarrow_1 \Uparrow_1$	290.27	17197	0.017	475
(i) JavaScript $\downarrow_2 \Downarrow_4$	312.14	21349	0.015	916
(i) TypeScript $\downarrow_2 \Downarrow_2$	315.10	21686	0.015	915
(c) Go $\uparrow_3 \Uparrow_{13}$	636.71	16292	0.039	228
(i) Jruby $\uparrow_2 \Downarrow_3$	720.53	19276	0.037	1671
(i) Ruby $\Uparrow_5$	855.12	26634	0.032	482
(i) PHP $\Uparrow_3$	1,397.51	42316	0.033	786
(i) Python $\Uparrow_{15}$	1,793.46	45003	0.040	275
(i) Lua $\downarrow_1$	2,452.04	209217	0.012	1961
(i) Perl $\uparrow_1$	3,542.20	96097	0.037	2148
(c) Swift	n.e.			

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Illustration de l'influence du contexte : le langage de programmation et la consommation - Résultats

binary-trees	Energy	Time	Ratio	Mb
(c) C	39.80	1125	0.035	131
(c) C++	41.23	1129	0.037	132
(c) Rust ⇓2	49.07	1263	0.039	180
(c) Fortran ⇑1	69.82	2112	0.033	133
(c) Ada ⇓1	95.02	2822	0.034	197
(c) Ocaml ↓1 ⇑2	100.74	3525	0.029	148
(v) Java ↑1 ⇓16	111.84	3306	0.034	1120
(v) Lisp ↓3 ⇓3	149.55	10570	0.014	373
(v) Racket ↓4 ⇓6	155.81	11261	0.014	467
(i) Hack ↑2 ⇓9	156.71	4497	0.035	502
(v) C# ↓1 ⇓1	189.74	10797	0.018	427
(v) F# ↓3 ⇓1	207.13	15637	0.013	432
(c) Pascal ↓3 ⇑5	214.64	16079	0.013	256
(c) Chapel ↑5 ⇑4	237.29	7265	0.033	335
(v) Erlang ↑5 ⇑1	266.14	7327	0.036	433
(c) Haskell ↑2 ⇓2	270.15	11582	0.023	494
(i) Dart ↓1 ⇑1	290.27	17197	0.017	475
(i) JavaScript ↓2 ⇓4	312.14	21349	0.015	916
(i) TypeScript ↓2 ⇓2	315.10	21686	0.015	915
(c) Go ↑3 ⇑13	636.71	16292	0.039	228
(i) Jruby ↑2 ⇓3	720.53	19276	0.037	1671
(i) Ruby ⇑5	855.12	26634	0.032	482
(i) PHP ⇑3	1,397.51	42316	0.033	786
(i) Python ⇑15	1,793.46	45003	0.040	275
(i) Lua ↓1	2,452.04	209217	0.012	1961
(i) Perl ↑1	3,542.20	96097	0.037	2148
(c) Swift	n.e.			

regex-redux	Energy	Time	Ratio	Mb
(i) TypeScript ↓2 ⇓12	24.65	2009	0.012	587
(c) C ↑1	24.83	805	0.031	151
(i) JavaScript ↓2 ⇓9	25.68	2096	0.012	525
(i) PHP ↑2 ⇓1	34.57	1667	0.021	182
(c) Pascal ↓1 ⇑4	35.20	2282	0.015	106
(i) Hack ↑2 ⇓2	38.96	2052	0.019	268
(c) Rust	40.26	2287	0.018	218
(c) Chapel ⇓11	97.19	4534	0.021	1055
(c) Ada ⇑5	148.66	5157	0.029	157
(i) Python ↓1	161.62	7116	0.023	429
(c) Ocaml ↓3 ⇓6	172.43	12978	0.013	948
(c) C++ ↓1 ⇑6	176.24	10656	0.017	216
(i) Ruby ↓4 ⇑4	192.88	14282	0.014	305
(v) Java ↑4 ⇓6	194.65	5694	0.034	1225
(i) Dart ↓1 ⇑4	197.92	13485	0.015	459
(i) Perl ↑4 ⇑13	236.24	7164	0.033	154
(i) Jruby ↑2 ⇓4	348.44	13477	0.026	1369
(v) Racket ↓1	358.20	26152	0.014	983
(v) C# ↑1 ⇑3	522.59	14723	0.035	851
(c) Swift ⇑5	538.11	41703	0.013	677
(v) F# ⇑7	650.51	46905	0.014	667
(c) Haskell	n.a.			
(c) Fortran	n.a.			
(c) Go	n.e.			
(i) Lua	n.e.			
(v) Erlang	n.a.			
(v) Lisp	n.e.			

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Illustration de l'influence du contexte : le langage de programmation et la consommation - Résultats

binary-trees	Energy	Time	Ratio	Mb
(c) C	39.80	1125	0.035	131
(c) C++	41.23	1129	0.037	132
(c) Rust ⇓$_2$	49.07	1263	0.039	180
(c) Fortran ⇑$_1$	69.82	2112	0.033	133
(c) Ada ⇓$_1$	95.02	2822	0.034	197
(c) Ocaml ↓$_1$ ⇑$_2$	100.74	3525	0.029	148
(v) Java ↑$_1$ ⇓$_{16}$	111.84	3306	0.034	1120
(v) Lisp ↓$_3$ ⇓$_3$	149.5			
(v) Racket ↓$_4$ ⇓$_6$	155.8			
(i) Hack ↑$_2$ ⇓$_9$	156.7			
(v) C# ↓$_1$ ⇓$_1$	189.7			
(v) F# ↓$_3$ ⇓$_1$	207.1			
(c) Pascal ↓$_3$ ⇑$_5$	214.6			
(c) Chapel ↑$_5$ ⇑$_4$	237.2			
(v) Erlang ↑$_5$ ⇑$_1$	266.1			
(c) Haskell ↑$_2$ ⇓$_2$	270.1			
(i) Dart ↓$_1$ ⇑$_1$	290.2			
(i) JavaScript ↓$_2$ ⇓$_4$	312.1			
(i) TypeScript ↓$_2$ ⇓$_2$	315.16	21886	0.015	915
(c) Go ↑$_3$ ⇑$_{13}$	636.71	16292	0.039	228
(i) Jruby ↑$_2$ ⇓$_3$	720.53	19276	0.037	1671
(i) Ruby ⇑$_5$	855.12	26634	0.032	482
(i) PHP ⇑$_3$	1,397.51	42316	0.033	786
(i) Python ⇑$_{15}$	1,793.46	45003	0.040	275
(i) Lua ↓$_1$	2,452.04	209217	0.012	1961
(i) Perl ↑$_1$	3,542.20	96097	0.037	2148
(c) Swift	n.e.			

regex-redux	Energy	Time	Ratio	Mb
(i) TypeScript ↓$_2$ ⇓$_{12}$	24.65	2009	0.012	587
(c) C ↑$_1$	24.83	805	0.031	151
(i) JavaScript ↓$_2$ ⇓$_9$	25.68	2096	0.012	525
(i) PHP ↑$_2$ ⇓$_1$	34.57	1667	0.021	182
(c) Pascal ↓$_1$ ⇑$_4$	35.20	2282	0.015	106
(i) Hack ↑$_2$ ⇓$_2$	38.96	2052	0.019	268
(c) Rust	40.26	2287	0.018	218
	97.19	4534	0.021	1055
	48.66	5157	0.029	157
	61.62	7116	0.023	429
	72.43	12978	0.013	948
	76.24	10656	0.017	216
	92.88	14282	0.014	305
	94.65	5694	0.034	1225
	97.92	13485	0.015	459
	36.24	7164	0.033	154
	48.44	13477	0.026	1369
	58.20	26152	0.014	983
(v) C# ↑$_1$ ⇑$_3$	522.59	14723	0.035	851
(c) Swift ⇑$_5$	538.11	41703	0.013	677
(v) F# ⇑$_7$	650.51	46905	0.014	667
(c) Haskell	n.a.			
(c) Fortran	n.a.			
(c) Go	n.e.			
(i) Lua	n.e.			
(v) Erlang	n.a.			
(v) Lisp	n.e.			

Deux benchmarks (contextes) deux résultats très différents ! ⇒ influence des tâches effectuées et du contexte

Illustration de l'influence du contexte : le langage de programmation et la consommation - Résultats

Le tableau ici est la normalisation sur la moyenne des benchmarks évoqués plus tôt. Il n'est ainsi pas représentatif d'un corpus de projets "génériques" ou "industriels" ⇒ **ne pas le sortir de son contexte !** (et ainsi éviter le café du commerce...)

	Energy
(c) C	1.00
(c) Rust	1.03
(c) C++	1.34
(c) Ada	1.70
(v) Java	1.98
(c) Pascal	2.14
(c) Chapel	2.18
(v) Lisp	2.27
(c) Ocaml	2.40
(c) Fortran	2.52
(c) Swift	2.79
(c) Haskell	3.10
(v) C#	3.14
(c) Go	3.23
(i) Dart	3.83
(v) F#	4.13
(i) JavaScript	4.45
(v) Racket	7.91
(i) TypeScript	21.50
(i) Hack	24.02
(i) PHP	29.30
(v) Erlang	42.23
(i) Lua	45.98
(i) Jruby	46.54
(i) Ruby	69.91
(i) Python	75.88
(i) Perl	79.58

	Time
(c) C	1.00
(c) Rust	1.04
(c) C++	1.56
(c) Ada	1.85
(v) Java	1.89
(c) Chapel	2.14
(c) Go	2.83
(c) Pascal	3.02
(c) Ocaml	3.09
(v) C#	3.14
(v) Lisp	3.40
(c) Haskell	3.55
(c) Swift	4.20
(c) Fortran	4.20
(v) F#	6.30
(i) JavaScript	6.52
(i) Dart	6.67
(v) Racket	11.27
(i) Hack	26.99
(i) PHP	27.64
(v) Erlang	36.71
(i) Jruby	43.44
(i) TypeScript	46.20
(i) Ruby	59.34
(i) Perl	65.79
(i) Python	71.90
(i) Lua	82.91

Illustration de l'influence du contexte : le langage de programmation et la consommation - Résultats

Le tableau ici est la normalisation sur la moyenne des benchmarks évoqués plus tôt. Il n'est ainsi pas représentatif d'un corpus de projets "génériques" ou "industriels"

⇒ **RQ2 Ici, les langages les plus rapides ne sont pas les plus économes !**

	Energy
(c) C	1.00
(c) Rust	1.03
(c) C++	1.34
(c) Ada	1.70
(v) Java	1.98
(c) Pascal	2.14
(c) Chapel	2.18
(v) Lisp	2.27
(c) Ocaml	2.40
(c) Fortran	2.52
(c) Swift	2.79
(c) Haskell	3.10
(v) C#	3.14
(c) Go	3.23
(i) Dart	3.83
(v) F#	4.13
(i) JavaScript	4.45
(v) Racket	7.91
(i) TypeScript	21.50
(i) Hack	24.02
(i) PHP	29.30
(v) Erlang	42.23
(i) Lua	45.98
(i) Jruby	46.54
(i) Ruby	69.91
(i) Python	75.88
(i) Perl	79.58

	Time
(c) C	1.00
(c) Rust	1.04
(c) C++	1.56
(c) Ada	1.85
(v) Java	1.89
(c) Chapel	2.14
(c) Go	2.83
(c) Pascal	3.02
(c) Ocaml	3.09
(v) C#	3.14
(v) Lisp	3.40
(c) Haskell	3.55
(c) Swift	4.20
(c) Fortran	4.20
(v) F#	6.30
(i) JavaScript	6.52
(i) Dart	6.67
(v) Racket	11.27
(i) Hack	26.99
(i) PHP	27.64
(v) Erlang	36.71
(i) Jruby	43.44
(i) TypeScript	46.20
(i) Ruby	59.34
(i) Perl	65.79
(i) Python	71.90
(i) Lua	82.91

Illustration de l'influence du contexte : le langage de programmation et la consommation - Résultats

Le tableau 5 du papier donne la vision en fonction du front de Pareto du choix des langages ⇒ **si un tableau devait être retenu, ce devrait être celui-ci !**

⇒ RQ3 Oui, on peut décider de manière automatique du langage à utiliser pour 1 critère mais pas pour plusieurs (temps, énergie, etc.)

Table 5. Pareto optimal sets for different combination of objectives.

Time & Memory	Energy & Time	Energy & Memory	Energy & Time & Memory
C • Pascal • Go	C	C • Pascal	C • Pascal • Go
Rust • C++ • Fortran	Rust	Rust • C++ • Fortran • Go	Rust • C++ • Fortran
Ada	C++	Ada	Ada
Java • Chapel • Lisp • Ocaml	Ada	Java • Chapel • Lisp	Java • Chapel • Lisp • Ocaml
Haskell • C#	Java	OCaml • Swift • Haskell	Swift • Haskell • C#
Swift • PHP	Pascal • Chapel	C# • PHP	Dart • F# • Racket • Hack • PHP
F# • Racket • Hack • Python	Lisp • Ocaml • Go	Dart • F# • Racket • Hack • Python	JavaScript • Ruby • Python
JavaScript • Ruby	Fortran • Haskell • C#	JavaScript • Ruby	TypeScript • Erlang
Dart • TypeScript • Erlang	Swift	TypeScript	Lua • JRuby • Perl
JRuby • Perl	Dart • F#	Erlang • Lua • Perl	
Lua	JavaScript	JRuby	
	Racket		
	TypeScript • Hack		
	PHP		
	Erlang		
	Lua • JRuby		
	Ruby		

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Économie d’énergie logicielle par un moyen simple dans la vie de tous les jours : l’Ad Blocker !

Green Computing | INFO4 | Quentin Perez | 2025-2026

L’ad blocker : un ami pour votre vie privée et pour la planète

Crawling parmis 5000 sites web les plus populaires de la liste Tranco. ⇒ 724 994 visites réelles sur des sites web.

Mesure de la consommation du navigateur via SmartWatts (voir CM4) sur 4 machines différentes avec 5 visites de chacun des sites (répétitions pour minimiser les erreurs statistiques)

Users Pay Twice: The Hidden Energy Cost of Web Advertising

Samuel Pélissier
CentraleSupélec, Inria, IRISA
Rennes, France

Naif Mehanna
Independent
Lille, France

Sterenn Roux
Univ. Rennes, CNRS, Inria, IRISA
Rennes, France

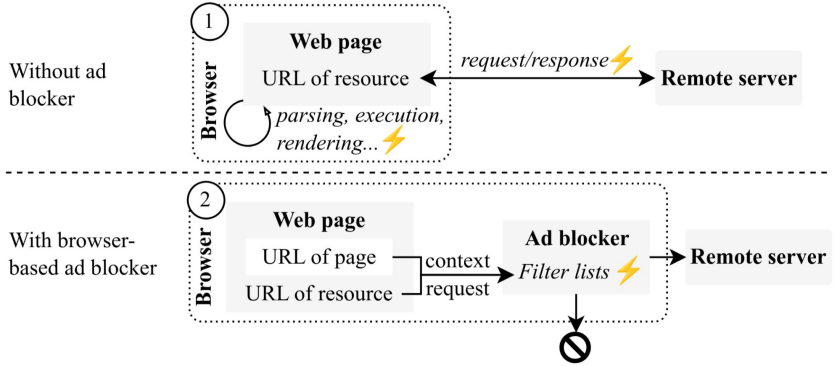
Quentin Perez
INSA Rennes, Univ. Rennes, CNRS, Inria, IRISA
Rennes, France

Walter Rudametkin
Univ. Rennes, CNRS, Inria, IRISA, IUF
Rennes, France

Johann Bourcier
Université de Pau et des Pays de l’Adour, E2S UPPA, LIUPPA
Anglet, France

Pierre Laperdrix
CNRS, Univ. Lille, Inria
Lille, France

[Pelissier26]



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

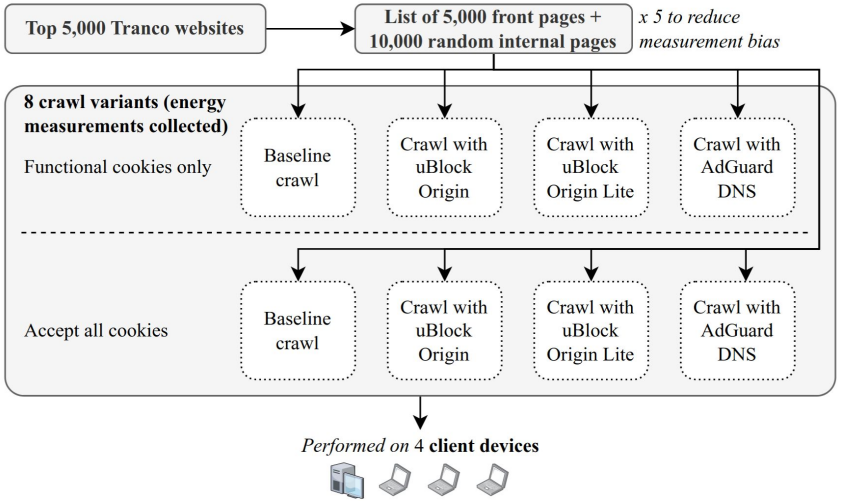
L’ad blocker : un ami pour votre vie privée et pour la planète

Crawling parmis 5000 sites web les plus populaires de la liste Tranco. ⇒ 724 994 visites réelles sur des sites web.

Mesure de la consommation du navigateur via SmartWatts (voir CM4) sur 4 machines différentes avec 5 visites de chacun des sites (répétitions pour minimiser les erreurs statistiques)

Visites selon différentes modalités :

- Acceptation ou non des cookies
- 2 Ad blockers intégrés dans le navigateur
  - uBlock Origin
  - uBlock Origin Lite
- 1 Ad blocker au niveau DNS : Ad Guard DNS



Device name	CPU	RAM (Go)	TDP (W)	Launch year
Intel NUC 9i7QNB	i7-9570H	32	45	2019
Dell Latitude 7490	i7-8650U	32	15	2017
Dell XPS 13 7390	i7-10510U	16	15	2019
Lenovo Thinkpad T480	i7-8550U	16	15	2017

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

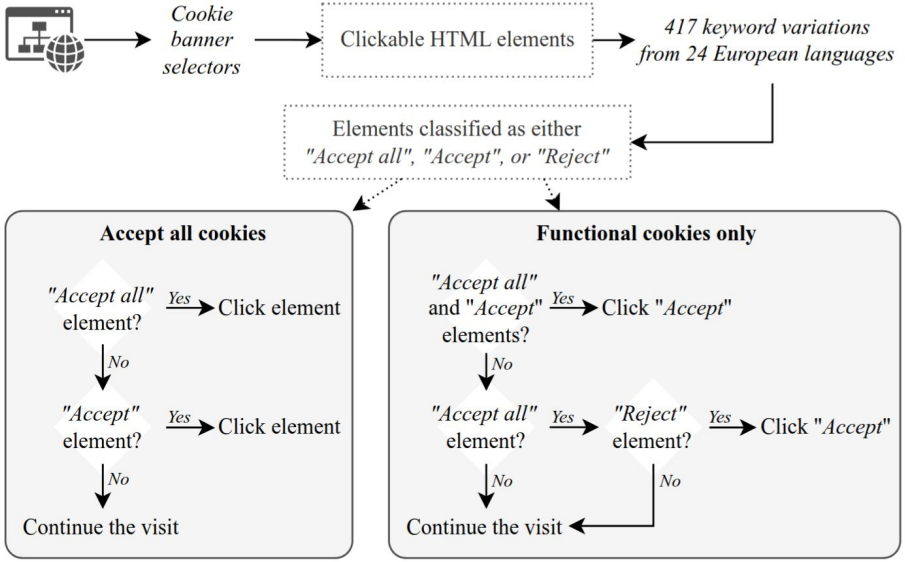
L’ad blocker : un ami pour votre vie privée et pour la planète

Crawling parmis 5000 sites web les plus populaires de la liste Tranco. ⇒ 724 994 visites réelles sur des sites web.

Mesure de la consommation du navigateur via SmartWatts (voir CM4) sur 4 machines différentes avec 5 visites de chacun des sites (répétitions pour minimiser les erreurs statistiques)

Visites selon différentes modalités :

- Acceptation ou non des cookies
- 2 Ad blockers intégrés dans le navigateur
  - uBlock Origin
  - uBlock Origin Lite
- 1 Ad blocker au niveau DNS : Ad Guard DNS



INSA | INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

L’ad blocker : un ami pour votre vie privée et pour la planète

- Cliquer sur « Tout accepter » sur les bannières de cookies augmente la consommation énergétique médiane de 2,6 %.

- Les sites web utilisant les enchères en temps réel (RTB) déclenchent en moyenne 8,5 requêtes HTTP supplémentaires lorsque les utilisateurs acceptent tous les cookies, dont 2 sont directement liées à la publicité et au suivi ⇒ 34 % d’augmentation de la consommation d'énergie côté client.

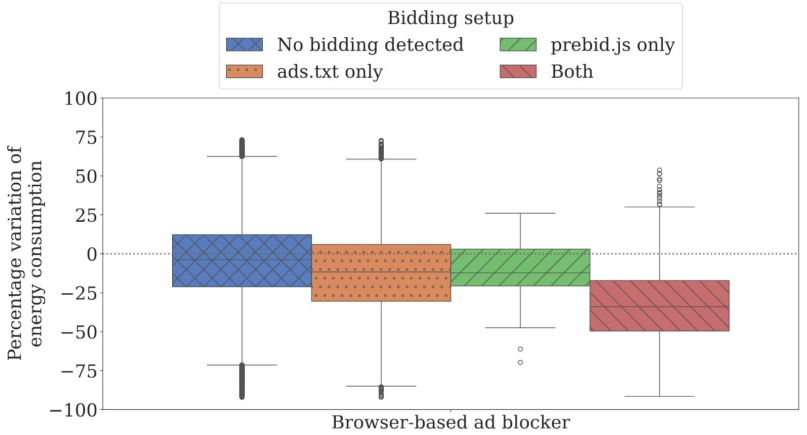


Figure 6: Energy consumption per header bidding setup type for the crawl variant "browser-based ad blocker with *Accept all cookies*", compared to the baseline crawl.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

L'ad blocker : un ami pour votre vie privée et pour la planète

- Cliquer sur « Tout accepter » sur les bannières de cookies augmente la consommation énergétique médiane de 2,6 %.

- Les sites web utilisant les enchères en temps réel (RTB) déclenchent en moyenne 8,5 requêtes HTTP supplémentaires lorsque les utilisateurs acceptent tous les cookies, dont 2 sont directement liées à la publicité et au suivi ⇒ 34 % d'augmentation de la consommation d'énergie côté client.

- Les bloqueurs de publicités réduisent la consommation d'énergie médiane de 9,6 %, malgré leur propre surcharge informatique.

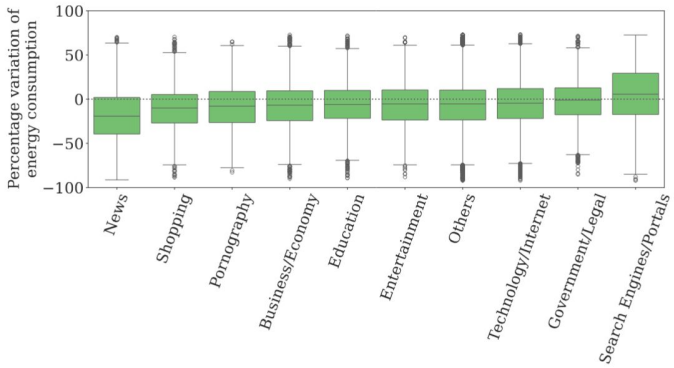


Figure 5: Energy consumption for the top 10 website categories of the crawl variant "browser-based ad blocker with *Accept all cookies*" compared to the baseline crawl.

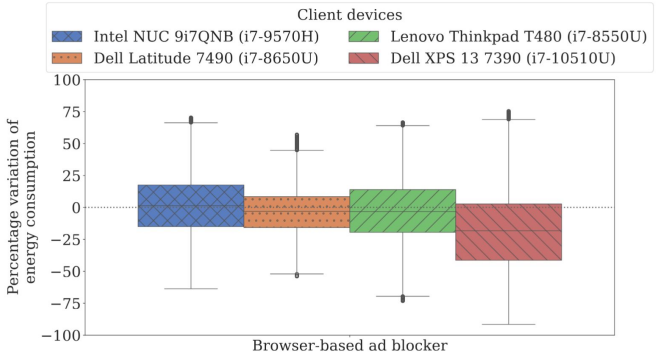


Figure 8: Energy consumption per client device for the crawl variant "browser-based ad blocker with *Accept all cookies*", compared to the baseline crawl.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

IT au sens large

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Référentiel GreenIT

Référentiel de 74 bonnes pratiques couvrant la plupart des domaines IT :

- Achat de matériel
- Mise en place d'équipements
- Recyclage
- Gouvernance et pilotage
- Centre IT
- etc.



[Référentiel PDF]

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Service numérique

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Référentiel Général d'Éco-Conception des Services Numériques

Référentiel produit par la Mission Interministérielle Numérique Écoresponsable couvrant la conception globale de services numériques :

- Stratégie
- Architecture
- UX/UI
- Frontend
- Backend
- Etc.



[Page du RGESN]

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

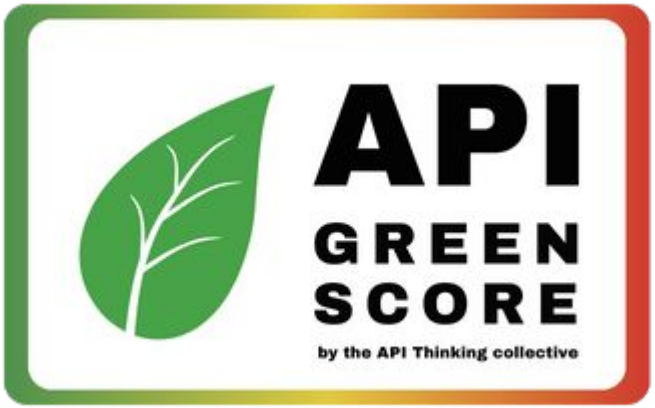
API Web

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

API Green Score

Référentiel de bonnes pratiques pour la conception d'API web

Permet l'évaluation de bonne conception de l'API à l'aide d'un score



[https://github.com/API-Green-Score/APIGreenScore?tab=readme-ov-file]

[https://www.collectif-api-thinking.com/assets/deliverables/worksites/50_CAT_API_Sustainable_IT.pdf]

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

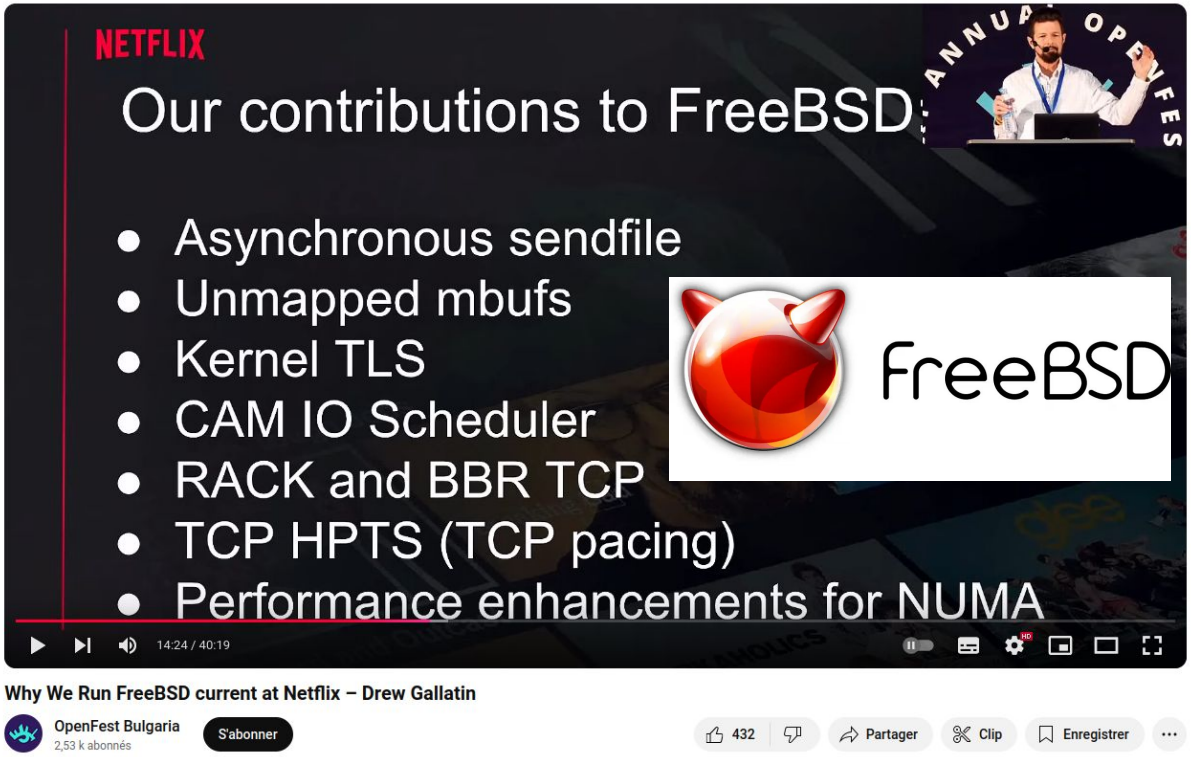
Optimisations au niveau de l'OS

Cas de Netflix : optimisations de FreeBSD

Participation au projet open-source FreeBSD :

- Création d'un "Internal FreeBSD"
- Optimisation de sécurité, spécialisation métier, **performances,** etc.

[https://www.youtube.com/watch?v=q4TZxj-Dq7s]

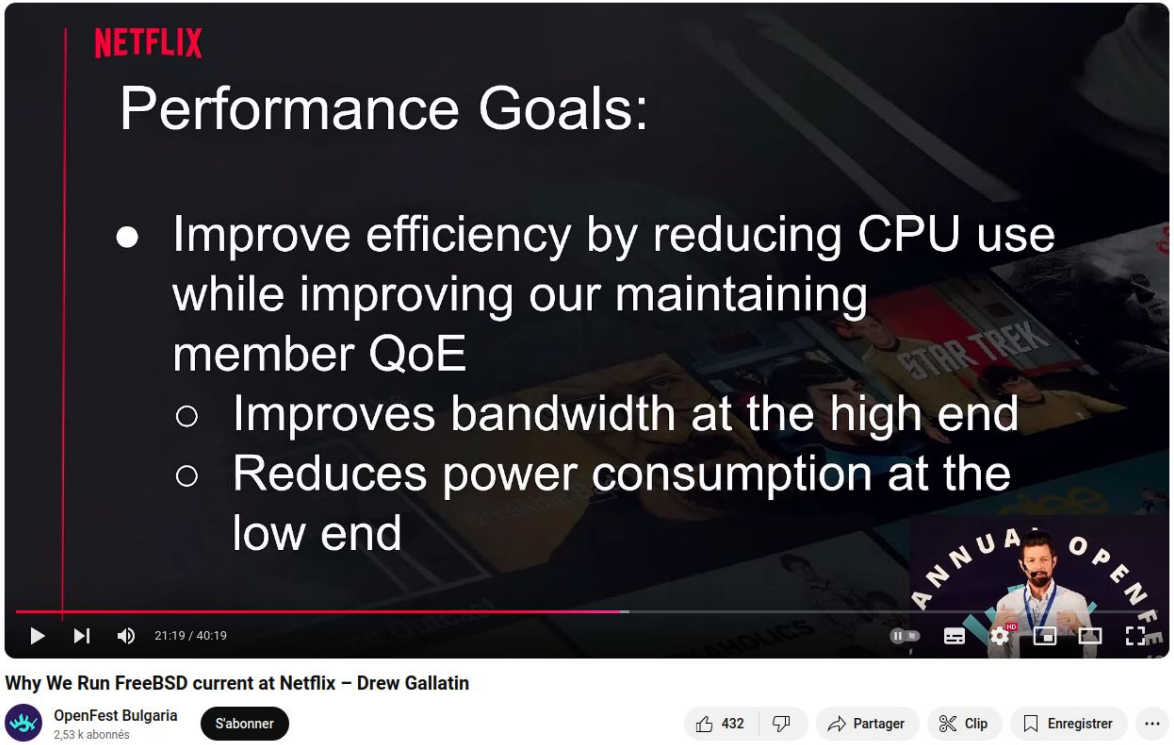


Cas de Netflix : optimisations de FreeBSD

Participation au projet open-source FreeBSD :

- Création d'un "Internal FreeBSD"
- Optimisation de sécurité, spécialisation métier, **performances,** etc.

[https://www.youtube.com/watch?v=q4TZxj-Dq7s]



Cas de Netflix : optimisations de FreeBSD

100W pour 100Gb/s

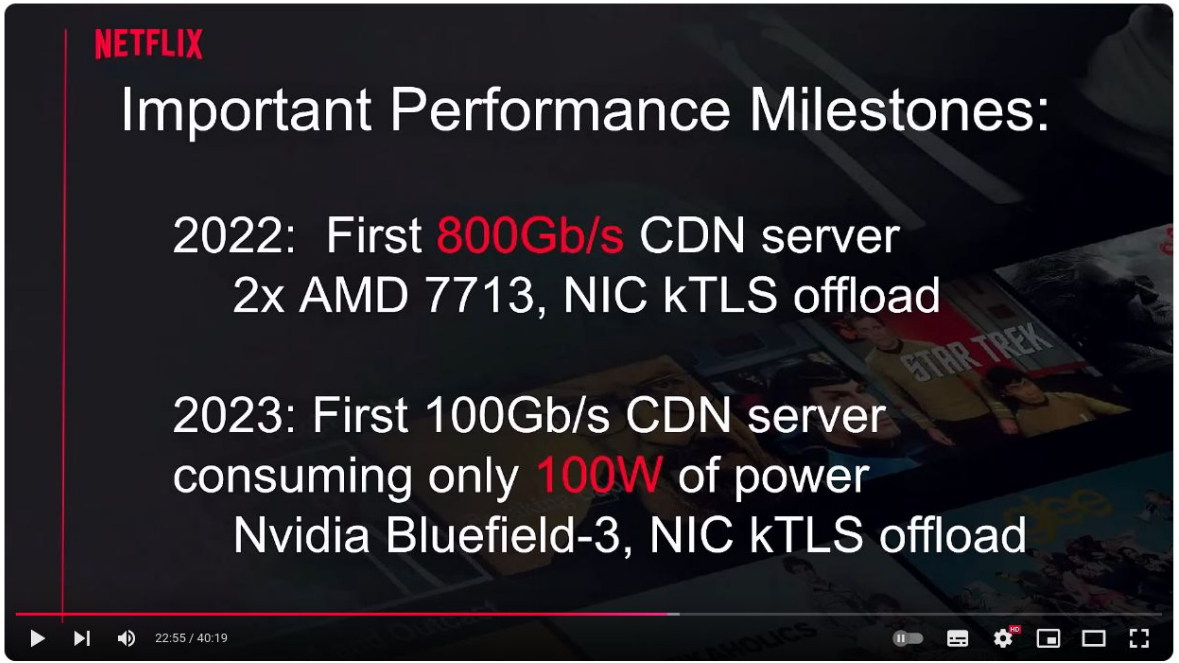
Un flux 4k = 0.035Gb/s

Calcul de coin de table :

100Gb/s / 0.035Gb/s = 2800 flux utilisateurs

100W / 2800 = 0.036W par flux !

[https://www.youtube.com/watch?v=q4TZxj-Dq7s]



Cas du noyau Linux: 30 lignes à impacts importants

- **30 lignes de codes au sein du noyau linux**
- Objectif : mieux gérer les interruptions et la charge du processeur en fonction du trafic réseau
- Augmentation de l'efficacité de l'application testée
- **Réduction la consommation d'énergie du datacenter jusqu'à 30 %.**

Kernel vs. User-Level Networking: Don't Throw Out the Stack with the Interrupts

PETER CAI, University of Waterloo, Canada
MARTIN KARSTEN, University of Waterloo, Canada

[Cai23]

```
1:  procedure EP_POLL(ep)                          ▷ epoll_wait() implementation
2:      avail ← are_events_available(ep)
3:      loop
4:          if avail then                                          ▷ Return path
5:              events ← get_events(ep)
6:              new_napi_id ← events[0].napi_id
7:              if new_napi_id ≠ ep.last_napi_id then
8:                  unmask_interrupts(ep.last_napi_id)
9:              end if
10:             ep.last_napi_id ← new_napi_id
11:             mask_interrupts(ep.last_napi_id)
12:             return events                                      ▷ Copy to user
13:         end if
14:         avail ← do_busy_poll(ep)
15:         if avail then
16:             continue
17:         end if
18:         unmask_interrupts(ep.last_napi_id)
19:         sleep_until_notified(ep)
20:         avail ← are_events_available(ep)
21:     end loop
22: end procedure
```

Fig. 4. Kernel Polling (Pseudo-code)

https://www.lemondeinformatique.fr/actualites/lire-une-modification-du-noyau-linux-rend-les-datacenters-plus-economes-en-energie-95910.html

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

IA et LLMs

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Empreinte écologique d'un LLM: BLOOM 176B

Étude de 2022 sur l'entraînement et l'inférence d'un modèle de 176M de paramètres

- Réalisé en France sur le super-calculateur Jean-Zay du CNRS
- 384 GPUs NVIDIA A100 SXM4, 80GB

ESTIMATING THE CARBON FOOTPRINT OF BLOOM, A 176B PARAMETER LANGUAGE MODEL

Alexandra Sasha Luccioni
Hugging Face
sasha.luccioni@hf.co

Sylvain Viguier
Graphcore
sylvainv@graphcore.ai

Anne-Laure Ligozat
LISN & ENSIIE
anne-laure.ligozat@lisn.upsaclay.fr

[Luccioni22]

Total training time	118 days, 5 hours, 41 min
Total number of GPU hours	1,082,990 hours
Total energy used	433,196 kWh
GPU models used	Nvidia A100 80GB
Carbon intensity of the energy grid	57 gCO_2eq/kWh

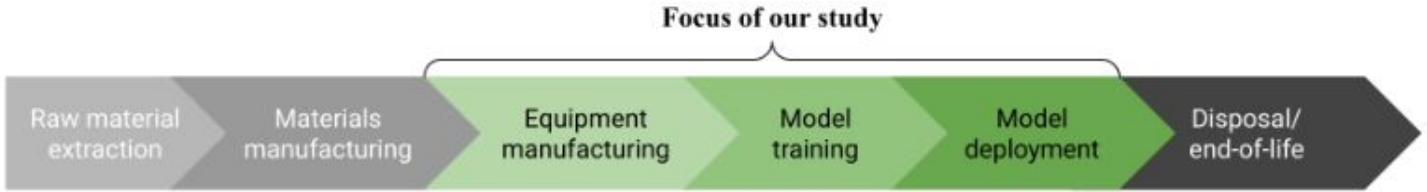
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Empreinte écologique d'un LLM: BLOOM 176B

Émissions CO2 liées au matériel :

11,2 tonnes CO2eq :

- Serveurs HPE Apollo 6500 Gen10 : 7,57t CO2eq
- GPUs A100 SMX4 : 3,64t CO2eq
- "Assuming a replacement rate of 6 years and 85% average usage"



[Luccioni22]

Empreinte écologique d'un LLM: BLOOM 176B

Sur une base 100, la consommation en mode idle + infrastructure représente 45,5%

Le reste de la consommation est effectué par du calcul.

Tests effectués en comparant des tâches calculatoires VS le même temps en idle.

Computing Mode	Power consumption	Percentage of total
Infrastructure consumption	27 kWh	13.5%
Idle consumption	64 kWh	32%
Dynamic consumption	109 kWh	54.5%
Total consumption	**200 kWh**	**100%**

Process	CO_2emissions (CO_2eq)	Percentage of total emissions
Embodied emissions	11.2 tonnes	22.2 %
Dynamic consumption	24.69 tonnes	48.9 %
Idle consumption	14.6 tonnes	28.9 %
Total	**50.5 tonnes**	**100.00%**

[Luccioni22]

Empreinte écologique d'un LLM: BLOOM 176B [Luccioni22]

Model name	Number of parameters	Datacenter PUE	Carbon intensity of grid used	Power consumption	CO_2eq emissions	CO_2eq emissions × PUE
GPT-3	175B	1.1	429 gCO_2eq/kWh	1,287 MWh	*502 tonnes*	552 tonnes
Gopher	280B	1.08	330 gCO_2eq/kWh	*1,066 MWh*	*352 tonnes*	380 tonnes
OPT	175B	*1.09*[2]	*231gCO_2eq/kWh*	*324 MWh*	70 tonnes	*76.3 tonnes*[3]
BLOOM	176B	1.2	57 gCO_2eq/kWh	433 MWh	25 tonnes	30 tonnes

Table 4: Comparison of carbon emissions between BLOOM and similar LLMs. Numbers in *italics* have been inferred based on data provided in the papers describing the models.

Pour l'entrainement des LLMs trois points sont à retenir :

- le PUE va avoir un effet multiplicateur sur les émissions
- **l'intensité carbone liée à la production électrique de la zone géographique (grille) a un impact très fort !**
- **Mais la taille des modèles également !**

Empreinte écologique d'un LLM: BLOOM 176B [Luccioni22]

Process	Energy consumed (kWh)	CO_2emissions (tonnes of CO_2eq)	Percentage of total emissions
176B BLOOM Model	433,196	24.69	37.24%
104B Model	266,522	15.19	22.92%
1B Model	158,972	9.06	13.68%
13B Model	87,210	4.97	7.49%
Other Models	64,257	3.66	5.53%
Miscellaneous Processes	57,961	3.3	4.98%
6B Model	51,686	2.95	4.45%
Model Evaluation	43,172	2.46	3.71%
Total	**1,163,088**	**66.29**	**100.00%**

Table 5: Breakdown of dynamic energy consumption and CO_2 emissions of different parts of the BigScience project

Pour l'entrainement des LLMs deux points sont à retenir :

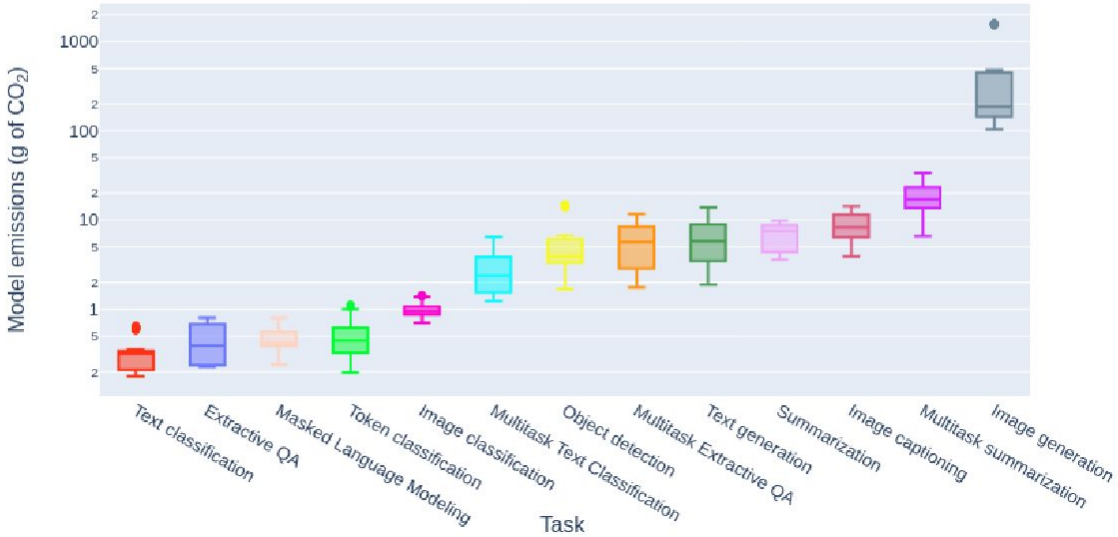
- le PUE va avoir un effet multiplicateur sur les émissions
- **l'intensité carbone liée à la production électrique de la zone géographique (grille) a un impact très fort !**
- **Mais la taille des modèles également !**

Coût des différentes tâches IA avec LLM

Power Hungry Processing: ⚡ Watts ⚡ Driving the Cost of AI Deployment?

ALEXANDRA SASHA LUCCIONI and YACINE JERNITE, Hugging Face, Canada/USA
EMMA STRUBELL, Carnegie Mellon University, Allen Institute for AI, USA

[Luccioni24]



Coût des différentes tâches IA avec LLM

Task	Datasets	Task	Datasets
image classification	CIFAR 10 [25] CIFAR 100 [25] ImageNet 1K [45]	**question answering**	SQuAD[44] SQuAD v2 [43] SciQ [23]
image captioning	Visual Genome [24] RedCaps [10] COCO [29]	**summarization**	SAMSum [15] CNN-Daily Mail [20] XSum [35]
image generation	DiffusionDB [54] ImageReward [58] SD Prompts [46]	**text classification**	IMDB [32] Rotten Tomatoes [39] SST 2 [48]
masked language modeling	BookCorpus [59] C4 [42] OSCAR [37]	**text generation**	WikiText [33] BookCorpus [59] OSCAR [37]
object detection	Visual Genome [24] CPPE-5 [9] COCO [29]	**token classification**	ReCoRD [53] WikiANN [38] CoNLL 2003 [50]

Table 1. A list of the tasks and datasets used in our study.

[Luccioni24]

	inference energy (kWh)	
task	**mean**	**std**
text classification	0.002	0.001
extractive QA	0.003	0.001
masked language modeling	0.003	0.001
token classification	0.004	0.002
image classification	0.007	0.001
object detection	0.038	0.02
text generation	0.047	0.03
summarization	0.049	0.01
image captioning	0.063	0.02
image generation	2.907	3.31

Table 2. Mean and standard deviation of energy per 1,000 queries for the ten tasks examined in our analysis.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Coût des différentes tâches IA avec LLM

[Luccioni24]

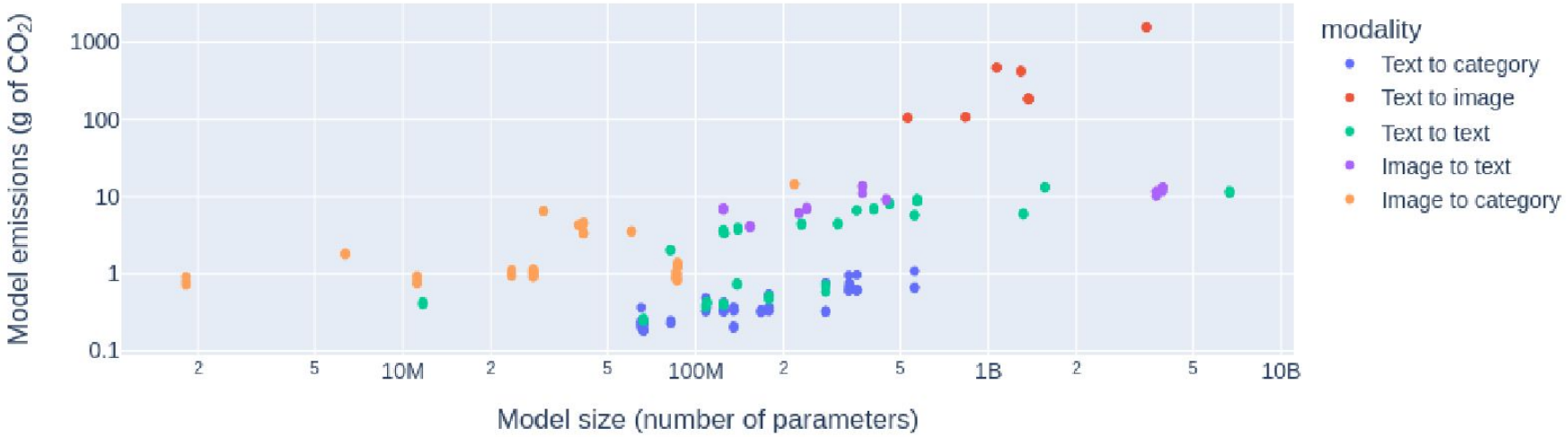


Fig. 2. The 5 modalities examined in our study, with the number of parameters of each model on the x axis and the average amount of carbon emitted for 1000 inferences on the y axis. NB: Both axes are in logarithmic scale.

Coût des différentes tâches IA avec LLM

[Luccioni24]

seq2seq models				decoder-only models			
model name	number of parameters	emissions (g CO_2eq)	energy (kWh)	model name	number of parameters	emissions (g CO_2eq)	energy (kWh)
Flan-T5-base	222M	3.67	0.026	**BLOOMz-560M**	559M	7.5	0.054
Flan-T5-large	750M	7.68	0.055	**BLOOMz-1B**	1.7B	8.66	0.062
Flan-T5-xl	2.8B	8.08	0.058	**BLOOMz-3B**	3B	10.17	0.073
Flan-T5-xxl	11B	11.48	0.083	**BLOOMz-7B**	7B	14.46	0.104

Table 3. Zero-shot models in our analysis with their architecture type, model size (in number of parameters), average quantity of emissions (in g of CO_2eq) and average energy usage (in kWh) for 1,000 inferences.

- **Influence du type de tâche sur la consommation à l'inférence**
- **La taille du modèle va influer sur la consommation**

Coût des différentes tâches IA avec LLM

[Luccioni24]

	BLOOMz-7B	BLOOMz-3B	BLOOMz-1B	BLOOMz-560M
Training energy (kWh)	51,686	25,634	17,052	10,505
Finetuning energy (kWh)	7,571	3,242	1,081	543
Inference energy (kWh)	1.0×10^{-4}	7.3×10^{-5}	6.2×10^{-5}	5.4×10^{-5}
Cost parity (# inferences)	592,570,000	395,602,740	292,467,741	204,592,592

Table 5. The BLOOMz models from our study with their training energy cost (from [31]), finetuning energy cost (from [34]), inference cost (from the present study), and cost parity, as the number of inferences required to sum to the training cost.

- **Influence du type de tâche sur la consommation à l'inférence**
- **La taille du modèle va influer sur la consommation**
- **Fine-tunner permet de limiter le cout énergétique**

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Optimisations simples du code : exemple avec Python

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Influence du type de boucle en Python

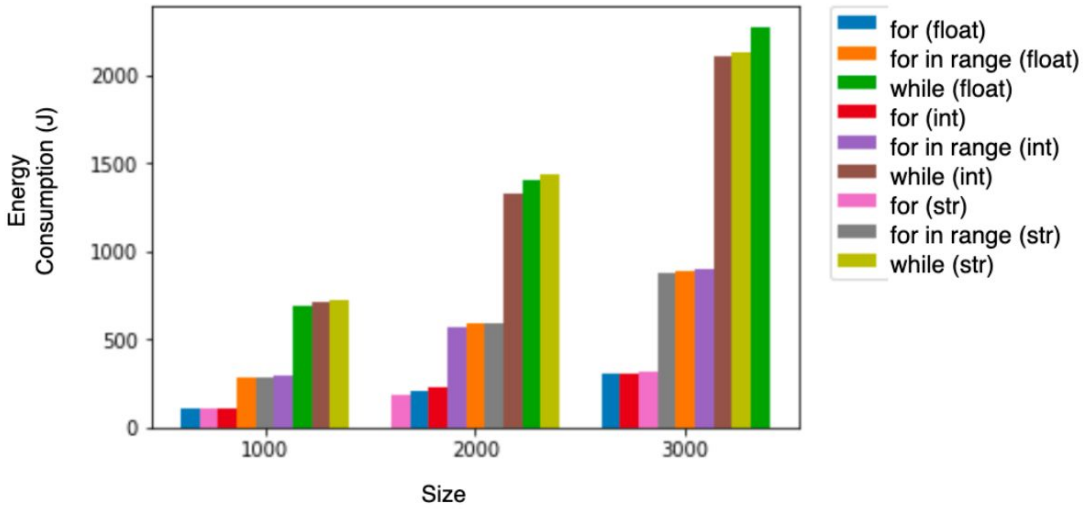
Comparaison de trois types de boucles :

```
for (i in range(len(n)))
```

```
for (element in collection)
```

```
while
```

⇒ Il faut privilégier les façons “Pythonic” de programmation à savoir éviter les while !



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Influence du type de boucle en Python

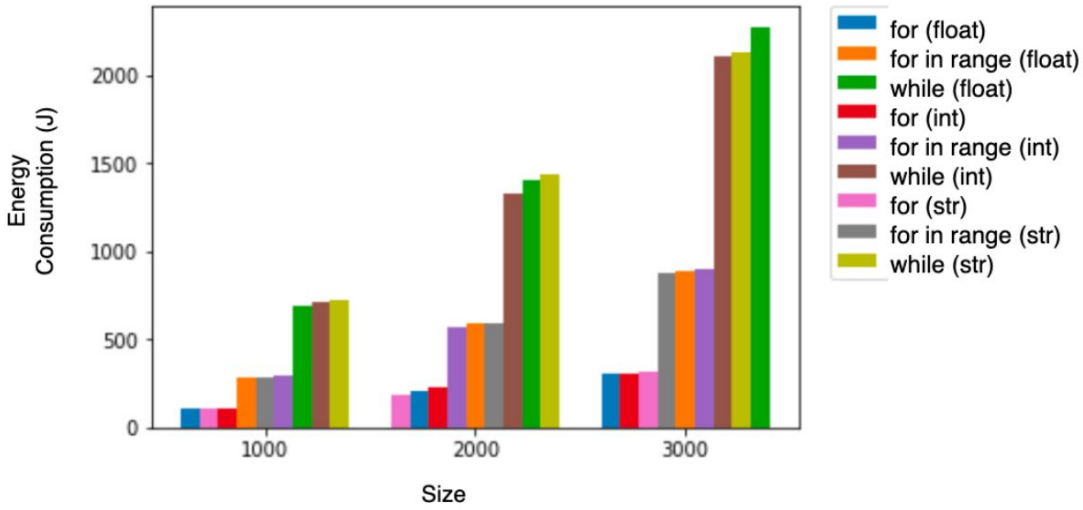
Comparaison de trois types de boucles :

```
for (i in range(len(n)))
```

```
for (element in collection)
```

```
while
```

Le while est toujours le plus consommateur ⇒ coût dû à l'incrément dans la boucle



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Influence du type de boucle en Python

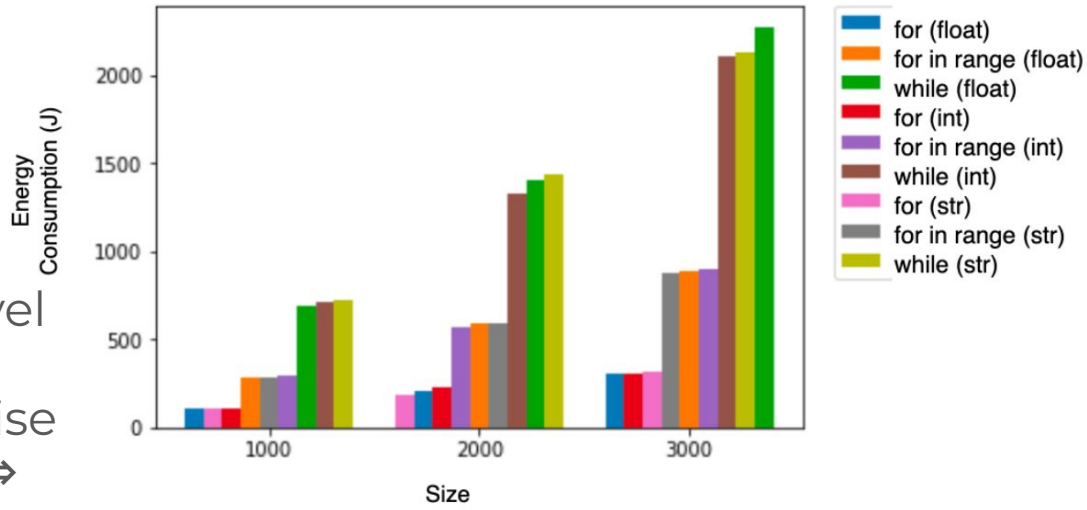
Comparaison de trois types de boucles :

```
for (i in range(len(n)))
```

```
for (element in collection)
```

```
while
```

Le “for in range” créé un nouvel itérateur à l’exécution pour chaque tour de boucle et utilise la fonction C range ($O(n)$=1)) ⇒ cout !



Influence du type de boucle en Python

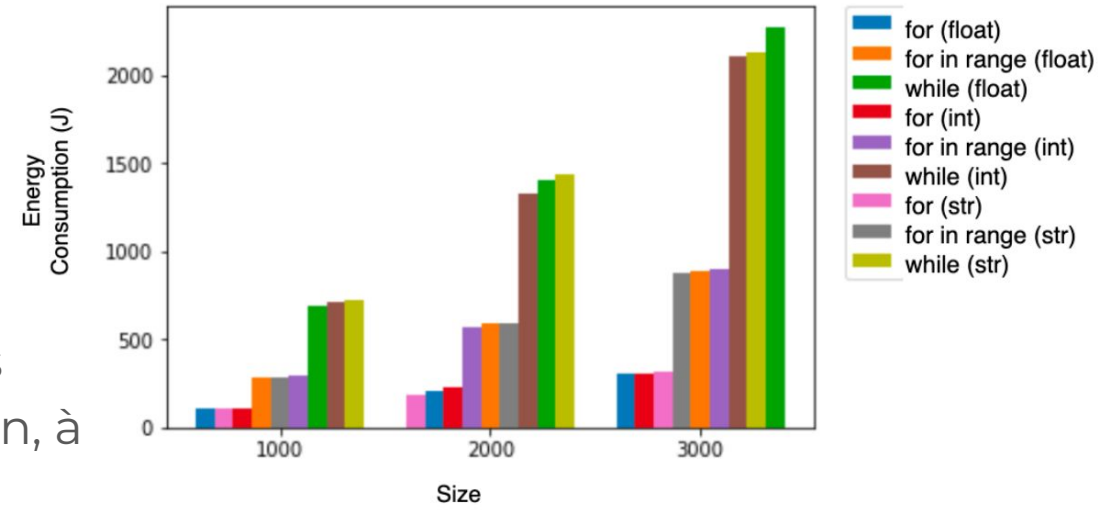
Comparaison de trois types de boucles :

```
for (i in range(len(n)))
```

```
for (element in collection)
```

```
while
```

⇒ Il faut privilégier les façons “Pythonic” de programmation, à savoir éviter les while !



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Build managers et impact énergétique

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Coût énergétique du build logiciel avec Maven et Gradle

Exécution de 3170 workflows Github Actions issus de projets Github programmés Java et utilisant Maven ou Gradle ⇒ **analyse finale sur 1167 workflows** :

- 739 utilisant Maven
- 427 utilisant Gradle
- 1 utilisant les 2

Collecte de l'énergie via SmartWatts pour monitorer les processsus liés au runner Github Action

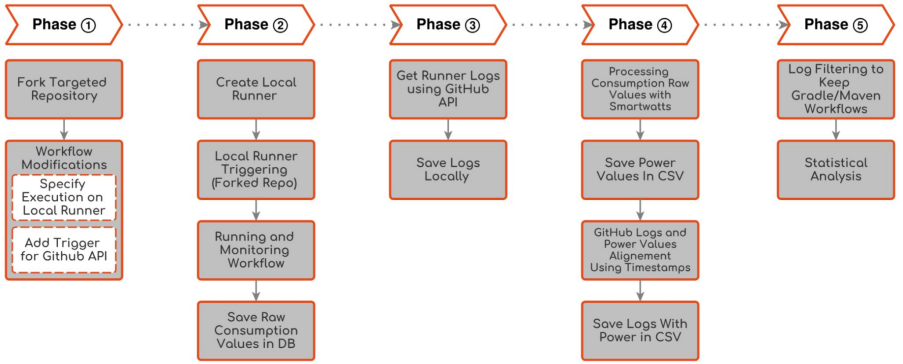
Evaluating the Energy Profile of Tasks Managed by Build Automation Tools in Continuous Integration Workflows: The Case of Apache Maven and Gradle

Sérgio Queiroz de Medeiros
Federal University of Rio Grande do Norte
Natal, Brazil
sergio.medeiros@ufrn.br

Romain Lefeuvre*, Benoit Combemale†
Univ. Rennes, IRISA, Inria
Rennes, France
romain.lefeuvre@irisa.fr*
benoit.combemale@irisa.fr†

Quentin Perez
INSA Rennes, Univ. Rennes, IRISA, Inria
Rennes, France
quentin.perez@irisa.fr

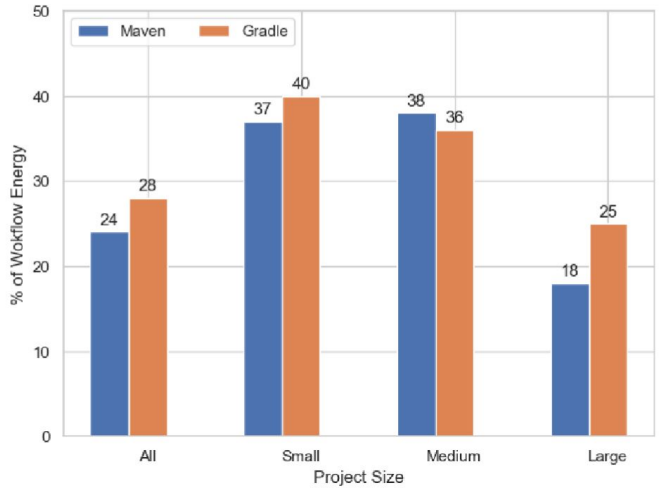
[Queiroz25]



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Coût énergétique du build logiciel avec Maven et Gradle

L'exécution de la construction du projet avec un build manager varie entre 18% et 40% du coût énergétique total du workflow suivant les tailles de projets.



(a) Workflows Tasks Managed by Maven or Gradle: Energy Share of Tasks Managed by Maven or Gradle Over Total Workflow Energy.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Coût énergétique du build logiciel avec Maven et Gradle

- L'exécution de la construction du projet avec un build manager varie entre 18% et 40% du coût énergétique total du workflow suivant les tailles de projets.
- Les tâches de :
  - Tests (unitaires et d'intégration)
  - Compilation
  - Documentation
  - Linting

⇒ sont les plus consommatrices en énergie pour la construction

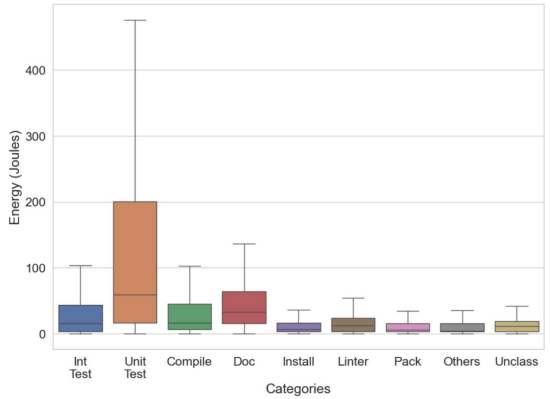


Fig. 7: Energy Consumption of Maven Tasks by Category.

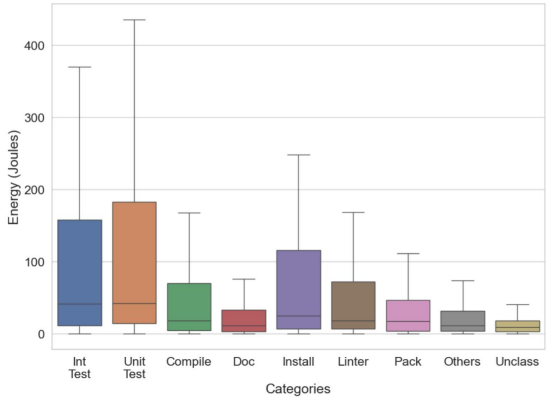


Fig. 9: Energy Consumption of Gradle Tasks by Category.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

Comparaison de la consommation CPU de différentes bibliothèques pour les entrées/sorties sur les fichiers :

- lectures/écritures
- fichiers de différentes tailles
- différents types de lecture/écriture (ex : lecture bufferisé ou non…)

Objectif :

Identifier la méthode la plus efficace pour l'ouverture de fichiers

Cas appliqué à un "contexte réel" !

Evaluating The Energy Consumption of Java I/O APIs

Zakaria OURNANI
Orange Labs/ Inria / Univ. Lille
France
zakaria.ournani@inria.fr

Romain ROUVOY
Univ. Lille / Inria / IUF
France
romain.rouvoy@univ-lille.fr

Pierre RUST
Orange Labs
France
pierre.rust@orange.com

Joel PENHOAT
Orange Labs
France
joel.penhoat@orange.com

[Ournani21]

TABLE I: The list of file sizes.

Method	File Size	Lines count	Line length
Tiny	100 KB	1,000	100
Small	15 MB	100,000	150
Medium	200 MB	1,000,000	200
Medium-large	3.2 GB	8,000,000	400
Large	16 GB	40,000,000	400

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

Comparaison de la consommation CPU de différentes bibliothèques pour les entrées/sorties sur les fichiers :

- lectures/écritures
- fichiers de différentes tailles
- différents types de lecture/écriture (ex : lecture bufferisé ou non...)

Objectif :

Identifier la méthode la plus efficace pour l'ouverture de fichiers

Cas appliqué à un "contexte réel" !

Class	Acronym	Method	Purpose	Availability
java.io.InputStreamReader	IOSTREAM	read(Byte[])	Read	JDK 1.0
java.io.OutputStreamWriter	IOSTREAM	write(String)	Write	JDK 1.0
java.io.FileInputStream	IOSTREAM	Skip(long) readAllBytes()	Seek ReadAll	JDK 1.0
java.io.BufferedInputStream	BIOSTREAM	readLine()	Read	JDK 1.0
java.io.BufferedOutpuStream	BIOSTREAM	write(String)	Write	JDK 1.0
java.io.FileReader	FILEREADER	read(char[])	Read	JDK 1.1
java.io.FileWriter	FILEWRITER	write(String,int,int)	Write	JDK 1.1
java.io.BufferedReader	BFILEREADER	readLine()	Read	JDK 1.1
java.io.BufferedWriter	BFILEWRITER	write(String)	Write	JDK 1.1
java.nio.channels.FileChannel	CHANNEL	read(ByteBuffer) write(ByteBuffer) position(long)	Read Write Seek	JDK 1.4
java.nio.FileChannel	OMCHANNEL	map(MapMode,long,long)	Read	JDK 1.4
java.nio.Files	NIOF	readLine() write(String) readAllLines(Path)	Read Write ReadAll	JDK 1.7
java.util.Scanner	SCANNER	nextLine()	Read	JDK 1.5
java.io.RandomAccesFile	RAF	readLine() writeBytes(String) seek(long)	Read Write Seek	JDK 1.0
apache.commons.io.FileUtils	APACHE	readFileToString(File,Charset) write(File,List¡String¿,Boolean) readFileToByteArray(File, Charset)	Read Write ReadAll	NA
google.common.io.CharSource google.common.io.CharSink google.common.io.Files	GUAVA	readLine() write(String) readLines(file, Charset)	Read Write ReadAll	NA

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

Comparaison de 9 méthodes de lecture.

Utiliser un Channel comparativement à d'autres méthodes permet d' économiser plus de 30% d'énergie comparativement à NIO.

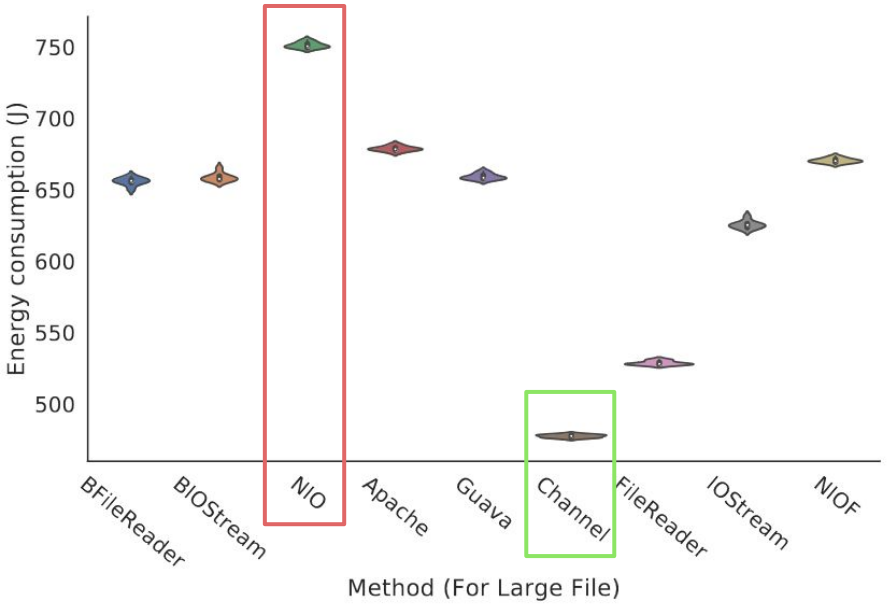


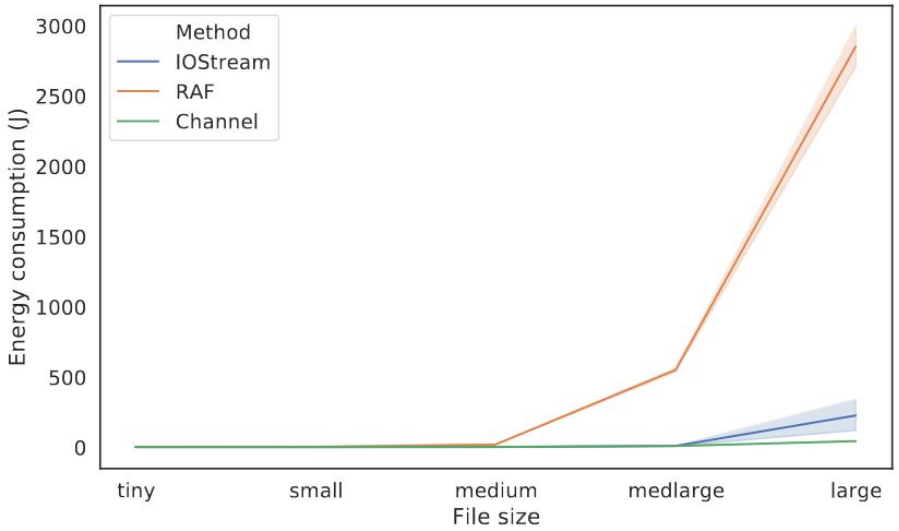
Fig. 3: Energy consumption of read methods for a large file.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

Comparaison de 9 méthodes de lecture.

Constance de la consommation des channels sur l'ensemble des tailles.



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

TABLE V: Energy consumption (joules) and execution time (ms) for writing files of different sizes by chunks.

Method	Tiny		Small		Medium		Medium-Large		Large	
	Energy	Time	Energy	Time	Energy	Time	Energy	Time	Energy	Time
Apache	0.20	13.3	8.4	760	8.7	$7.8 * 10^3$	679	$6.2 * 10^4$	2526	$2.3 * 10^5$
BFileWriter	0.24	15.1	9.5	794	8.6	$7.8 * 10^3$	$6.1 * 10^4$	$11.3 * 10^3$	2501	$2.3 * 10^5$
BIOStream	0.22	13.7	8.9	770	82	$7.5 * 10^3$	662	$6.0 * 10^4$	2502	$2.3 * 10^5$
Channel	0.23	14.5	10	919	99	$9.1 * 10^3$	798	$7.3 * 10^4$	3293	$2.8 * 10^5$
FileWriter	0.15	10.5	8.4	764	83	$7.6 * 10^3$	669	$6.1 * 10^4$	2518	$2.3 * 10^5$
Guava	1.18	80.7	95	6882	962	$7.1 * 10^4$	7507	$5.5 * 10^5$	15592	$1.7 * 10^6$
IOStream	0.12	9	8.5	765	83	$7.5 * 10^3$	671	$6.1 * 10^4$	2522	$2.3 * 10^5$
NIOF	0.32	22.3	27	1714	238	$1.5 * 10^4$	1897	$1.2 * 10^5$	3684	$3.4 * 10^5$
RAF	0.14	10.6	10	920	98	$9 * 10^3$	795	$7.3 * 10^4$	2932	$2.6 * 10^5$

Écrire un fichier consomme plus que le lire :

- 500 joules pour la lecture de fichier de grande taille dans le cas le plus efficace
- **x5 pour écrire** (dans le cas le plus favorable)

Guava pour tout type de fichier consomme 6x plus d'énergie que la meilleure solution

Channel ⇒ **le plus efficace sur tout type de lecture**, est **30% plus consommateur que la meilleure solution sur des fichiers larges.**

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliothèques I/O Java et consommation

Contrairement à la lecture, dans le cas de l'écriture, il n'y a pas de bibliothèque qui permet d'avoir une consommation constante sur la taille.

Influence de la taille sur la consommation à partir des fichiers de tailles médiums ⇒ Contexte !

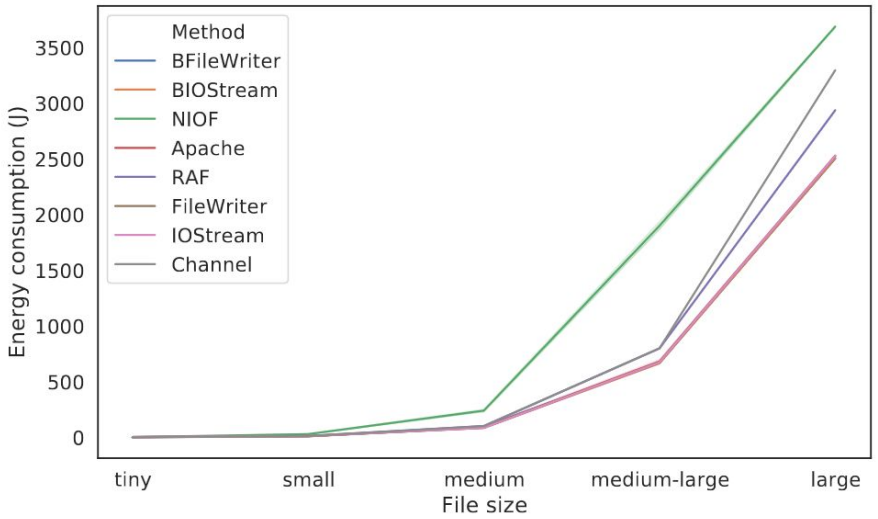


Fig. 7: Energy consumption to write files for several file sizes.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

Green Computing | INFO4 | Quentin Perez | 2025-2026

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

Comparaison de 12 distributions de JVM et évaluation sur un panel de 12 projets de natures variées pour minimiser les biais de spécificité

Evaluating the Impact of Java Virtual Machines on Energy Consumption

Zakaria Ournani
Orange Labs / Inria / Univ. Lille
zakaria.ournani@inria.fr

Mohammed Chakib Belgaid
Inria / Univ. Lille
chakib.belgaid@inria.fr

Romain Rouvoy
Univ. Lille / Inria / IUF
romain.rouvoy@univ-lille.fr

Pierre Rust
Orange Labs
pierre.rust@orange.com

Joël Penhoat
Orange Labs
joel.penhoat@orange.com

Distribution	Provider	Support	Selected versions
HotSpot	**Adopt OpenJDK**	All	8.0.275, 11.0.9, 12.0.2, 13.0.2, 14.0.2, 15.0.1
HotSpot	**Oracle**	All	8.0.265, 9.0.4, 10.0.2, 11.0.2, 12.0.2, 13.0.2, 14.0.2, 15.0.1, 16.ea.24
Zulu	**Azul Systems**	All	8.0.272, 9.0.7, 10.0.2, 11.0.9, 12.0.2, 13.0.5, 14.0.2, 15.0.1
SapMachine	**SAP**	All	11.0.9, 12.0.2, 13.0.2, 14.0.2, 15.0.1
Librca	**BellSoft**	All	8.0.275, 11.0.9, 12.0.2, 13.0.2, 14.0.2, 15.0.1
Corretto	**Amazon**	Mjr	8.0.275, 11.0.9, 15.0.1
HotSpot	**Trava OpenJDK**	LTS	8.0.232, 11.0.9
Dragonwell	**Alibaba**	LTS	8.0.272, 11.0.8
OpenJ9	**Eclipse**	All	8.0.275, 11.0.9, 12.0.2, 13.0.2, 14.0.2, 15.0.1
GraalVM	**Oracle**	LTS	19.3.4.r8, 19.3.4.r11, 20.2.0.r8, 20.2.0.r11
Mandrel	**Redhat**	LTS	20.2.0.0

[Ournani21]

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

Comparaison de 12 distributions de JVM et évaluation sur un panel de 12 projets de natures variées pour minimiser les biais de spécificité

Benchmark	Description	Focus
ALS	Factorize a matrix using the alternating least square algorithm on spark	Data-parallel, compute-bound
Avrora	Simulates and analyses for AVR microcontrollers	Fine-grained multi-threading, events queue
Dotty	Uses the dotty Scala compiler to compile a Scala codebase	Data structure, synchronization
Fj-Kmeans	Runs K-means algorithm using a fork-join framework	Concurrent data structure, task parallel
H2	Simulates an SQL database by executing a TPC-C like benchmark written by Apache	Query processing, transactions
Lusearch	Searches keywords over a corpus of data comprising the works of Shakespeare and the King James bible	Externally multi-threaded
Neo4j	Runs analytical queries and transactions on the Neo4j database	Query Processing, Transactions
Philosophers	Solves dining philosophers problem	Atomic, guarded blocks
PMD	Analyzes a list of Java classes for a range of source code problems	Internally multi-threaded
Reactors	Runs a set of message-passing workloads based on the reactors framework	Message-passing, critical-sections
Scrabble	Solves a scrabble puzzle using Java streams	Data-parallel, memory-bound
Sunflow	Renders a classic Cornell box; a simple scene comprising two teapots and two glass spheres within an illuminated box	Compute-bound

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

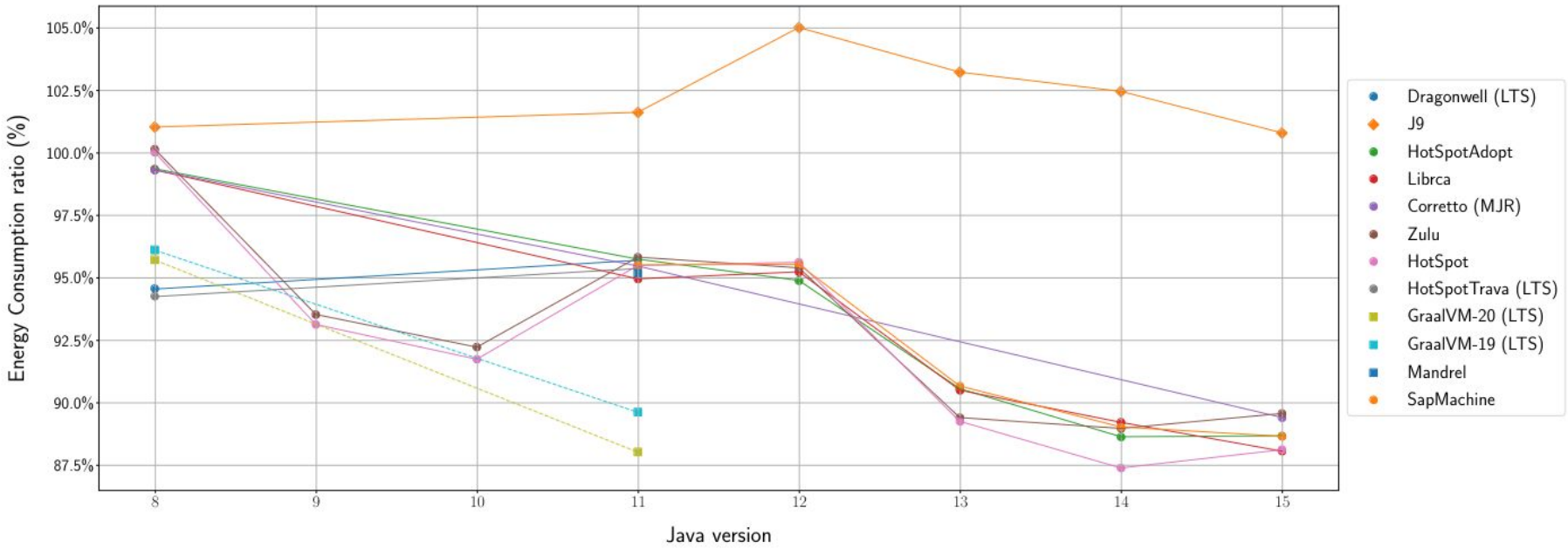


Figure 1: Energy consumption evolution of selected JVM distributions along versions.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

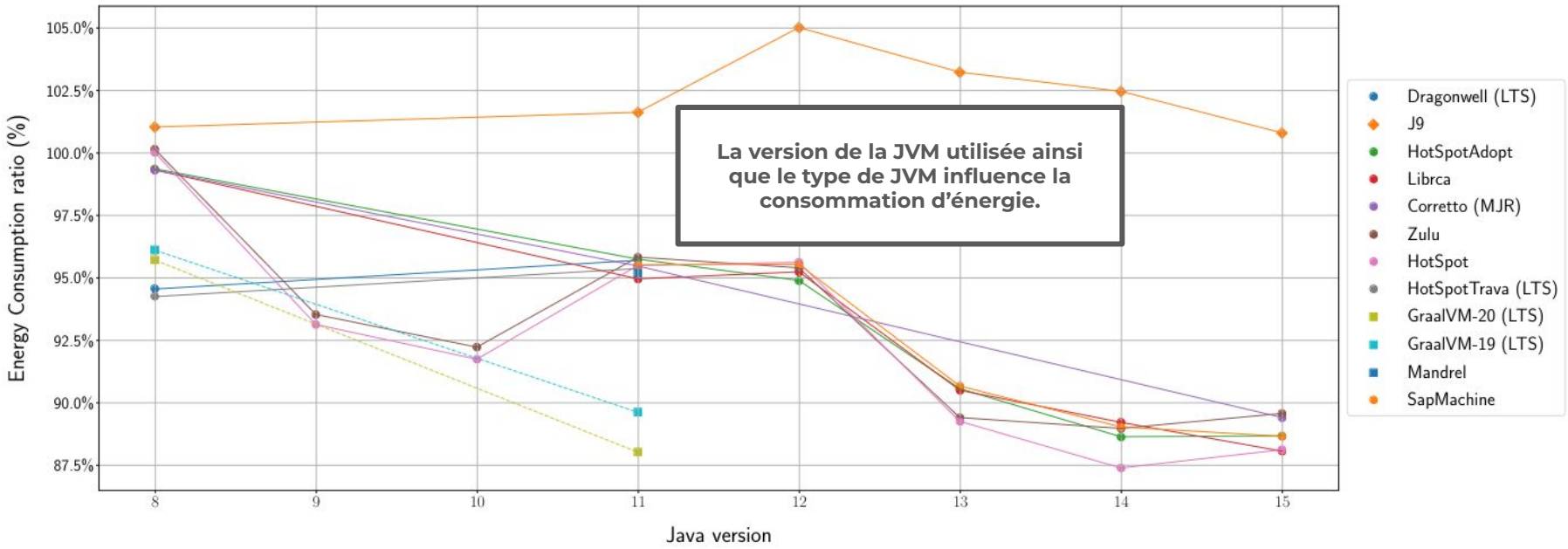


Figure 1: Energy consumption evolution of selected JVM distributions along versions.

Impact du type de JVM sur la consommation d'énergie

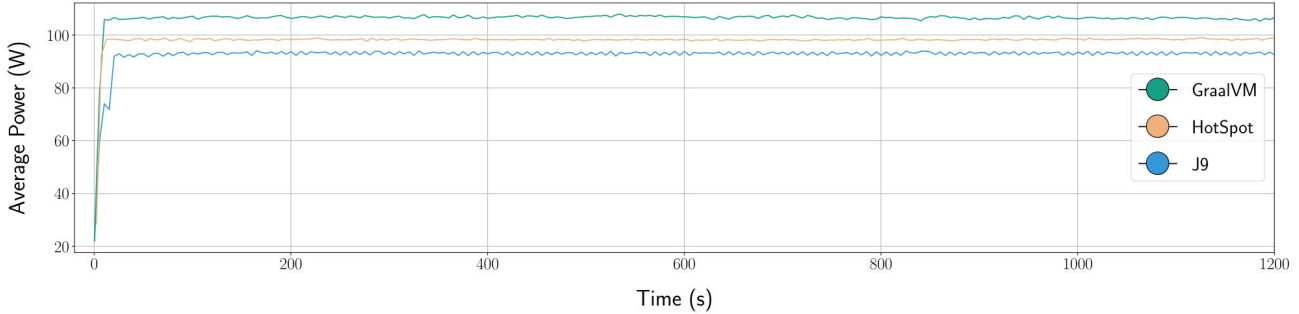
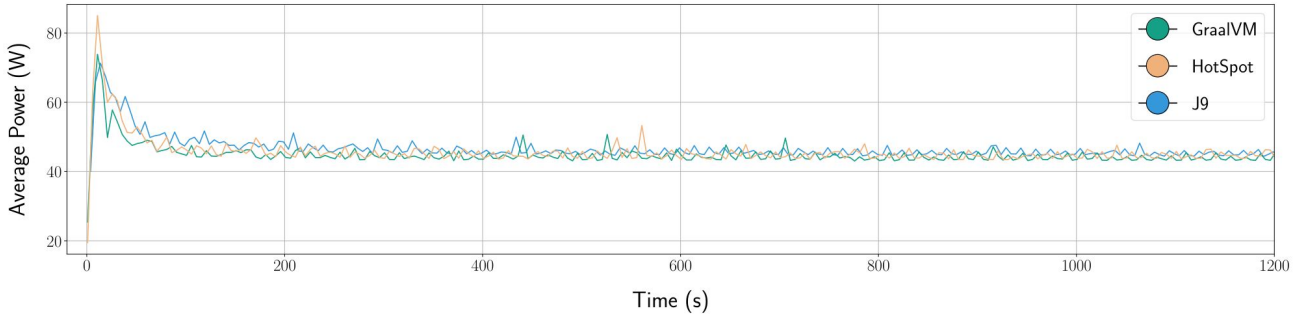


Figure 4: Power consumption of Scrabble as a service for HotSpot, GraalVM & J9.



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

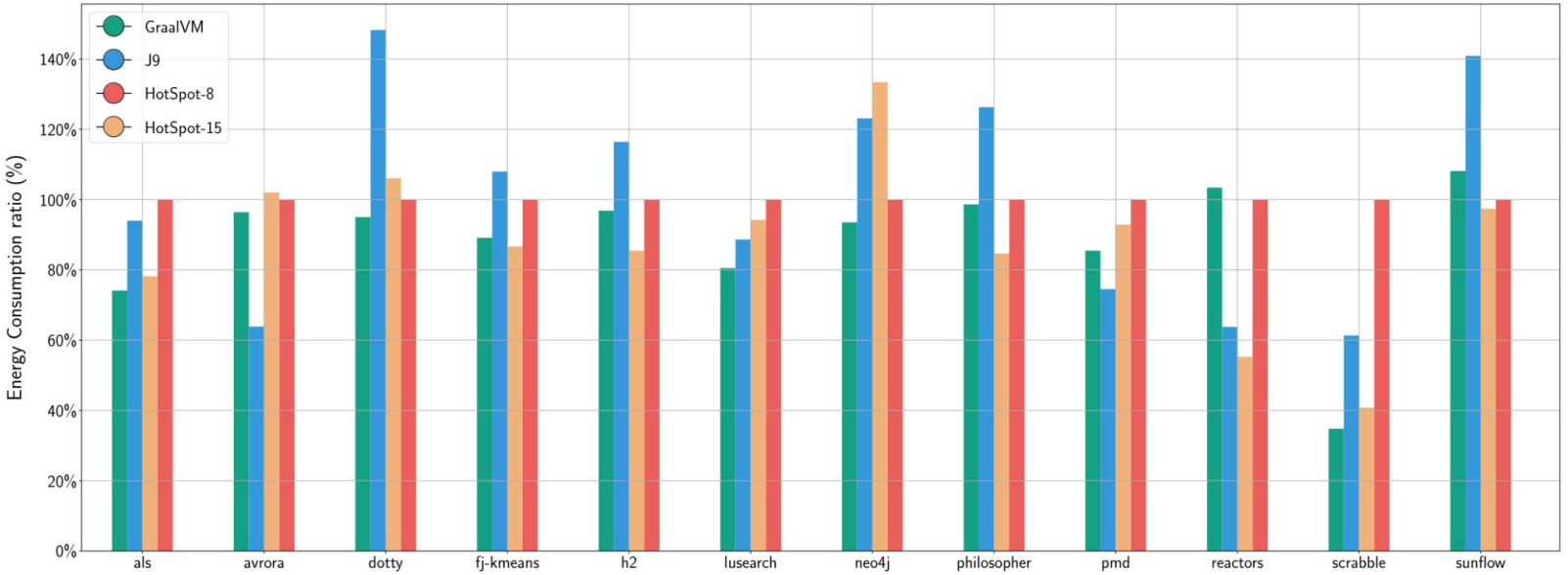


Figure 3: Energy consumption comparison across Java benchmarks for HotSpot, GraalVM & J9.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Impact du type de JVM sur la consommation d'énergie

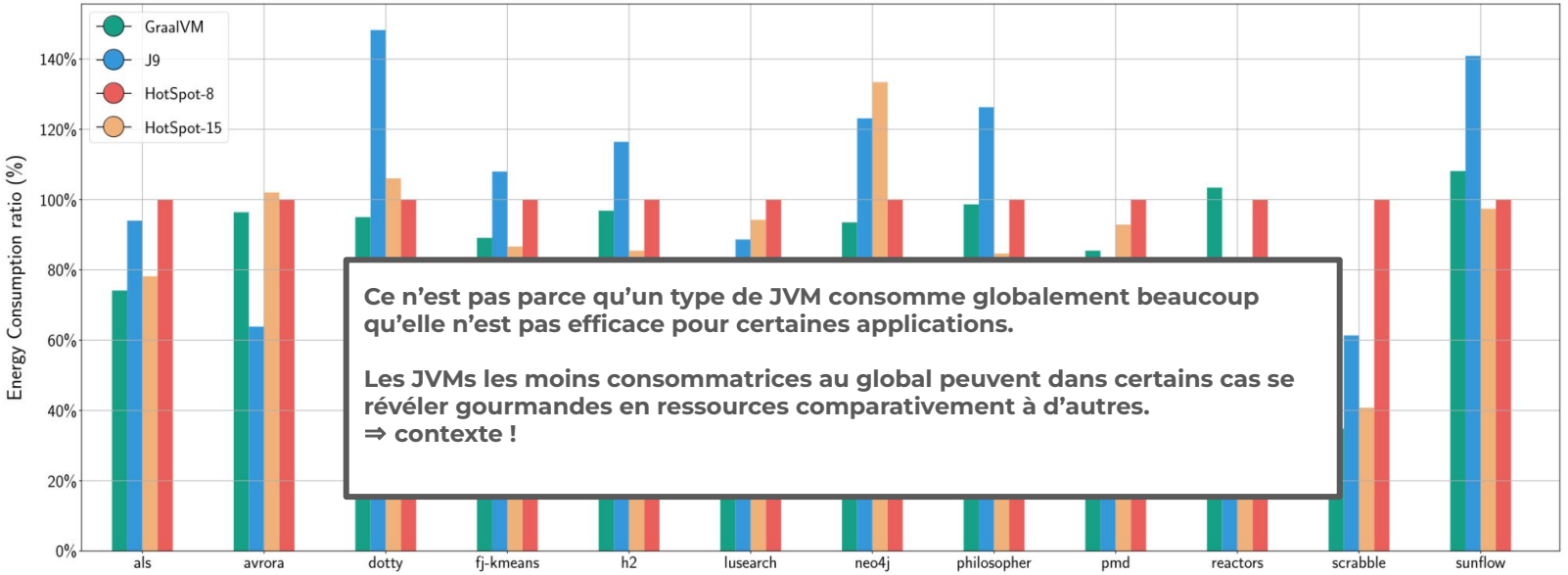


Figure 3: Energy consumption comparison across Java benchmarks for HotSpot, GraalVM & J9.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie

Green Computing | INFO4 | Quentin Perez | 2025-2026

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie

Intuition : utiliser Docker induit un coût non négligeable notamment lié à la conteneurisation et à la couche Docker

Objectif : étudier ce cout énergétique lié à l'utilisation de Docker

Comparaison en utilisant des applications "industrielles dockerisés"

How does docker affect energy consumption? Evaluating workloads in and out of Docker containers

Eddie Antonio Santos*, Carson McLean, Christopher Solinas, Abram Hindle

Department of Computing Science, University of Alberta, Edmonton, Canada

[Santos18]

Software	Version	Docker Image
Distribution	Ubuntu Server 16.04.1 LTS	
Kernel	Linux 4.4.0	
Docker	1.12.1	
Apache	2.4.10	php:5.6-apache
PHP	5.6.24	php:5.6-apache
MySQL	5.7.15	mysql:5.7.15
WordPress	4.6.0	wordpress:4.6-apache
Redis	3.2.3	redis:3.2.3
PostgreSQL	9.5.4	postgres:9.5.4

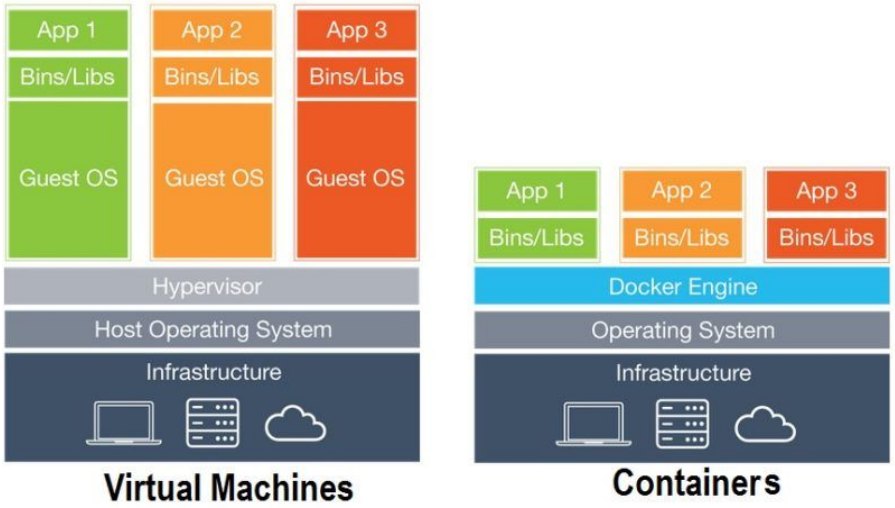
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie

Intuition : utiliser Docker induit un coût non négligeable notamment lié à la conteneurisation et à la couche Docker

Objectif : étudier ce cout énergétique lié à l'utilisation de Docker

Comparaison en utilisant des applications "industrielles dockerisés"

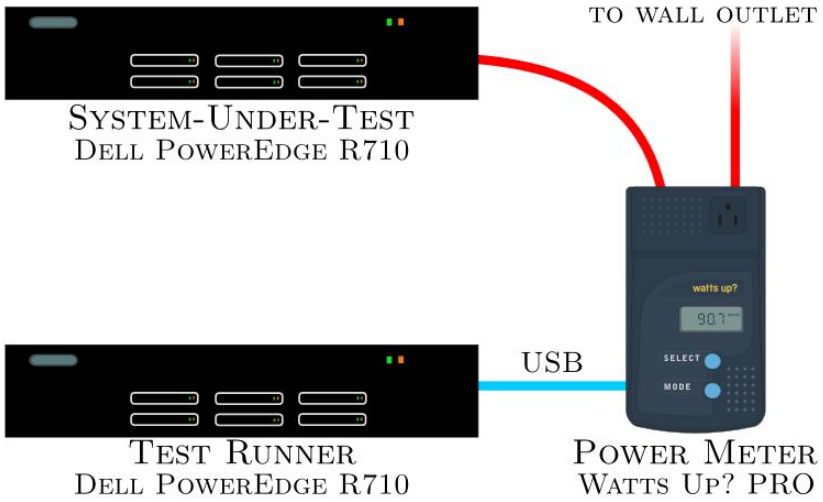


INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie

Comparaison en utilisant des applications "industrielles dockerisés"

Monitoring de la machine sous test avec un wattmètre physique relié en USB à une machine de mesure.



Hardware configuration of the System-Under-Test and the test runner.

CPU	2 × six-core Intel Xeon X5670 at 2.93 GHz
RAM	72 GiB ECC DDR3
Network	Gigabit Ethernet connection
Storage	146GB SAS hard drive at 15,000 RPM
Power supply	870 W (120 V ~ 12A at 60 Hz)

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie

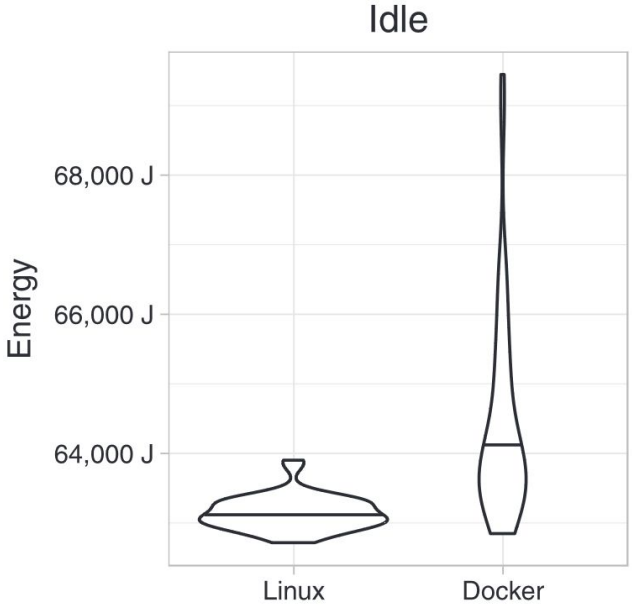
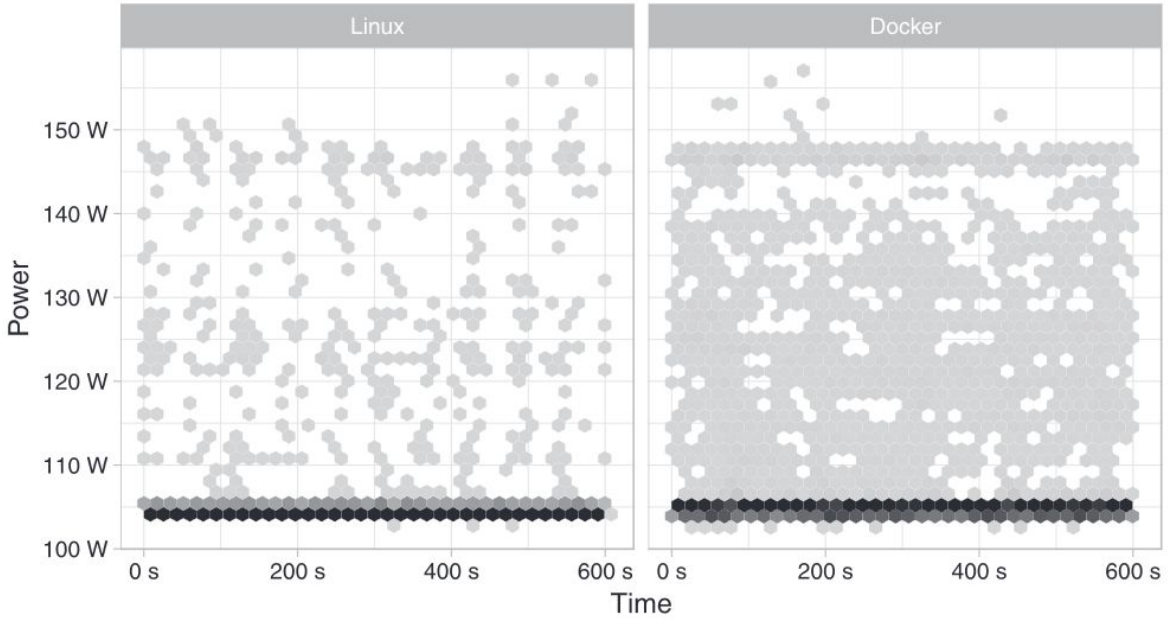


Fig. 2. Violin plot of idle energy consumption.



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie

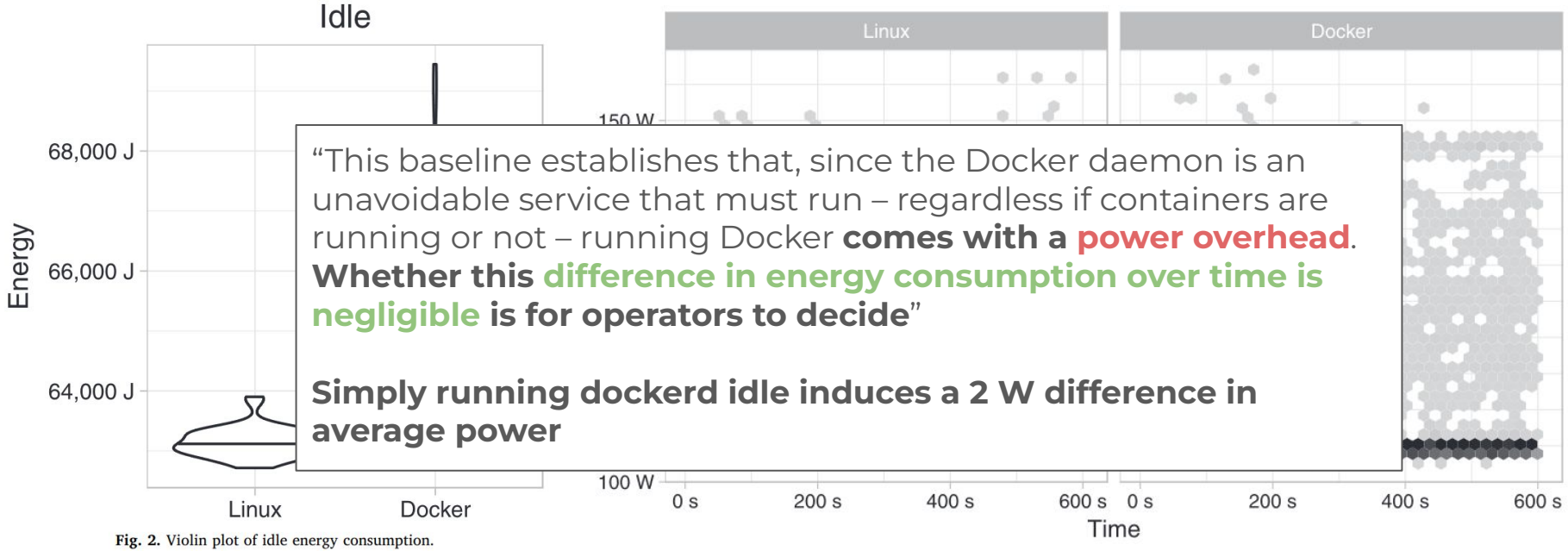


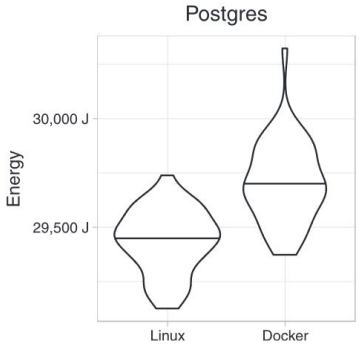
Fig. 2. Violin plot of idle energy consumption.

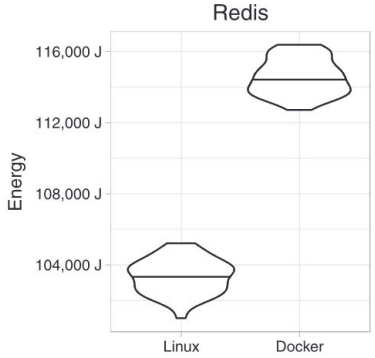
"This baseline establishes that, since the Docker daemon is an unavoidable service that must run – regardless if containers are running or not – running Docker **comes with a power overhead**. **Whether this difference in energy consumption over time is negligible is for operators to decide**"

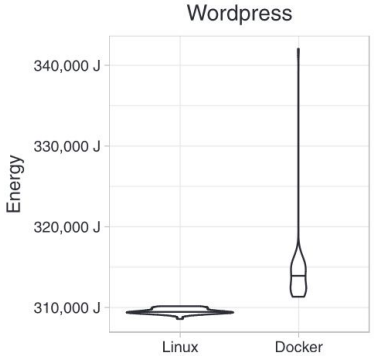
Simply running dockerd idle induces a 2 W difference in average power

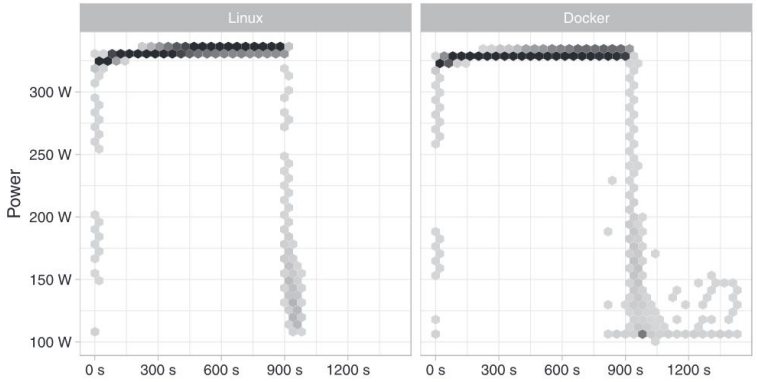
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

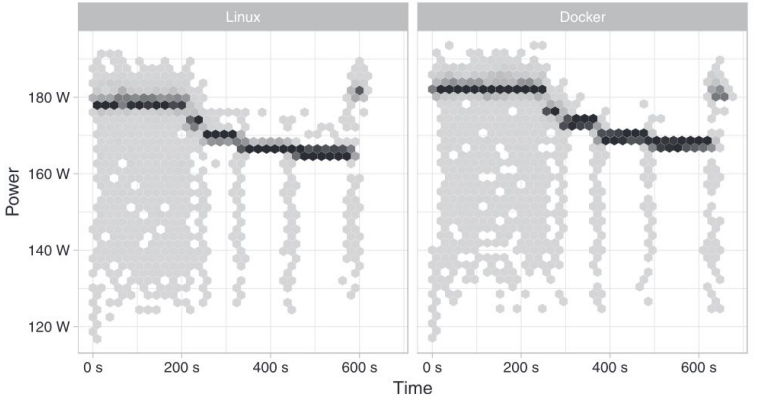
Utilisation de Docker et consommation d'énergie







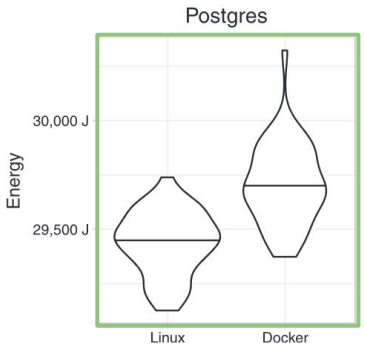


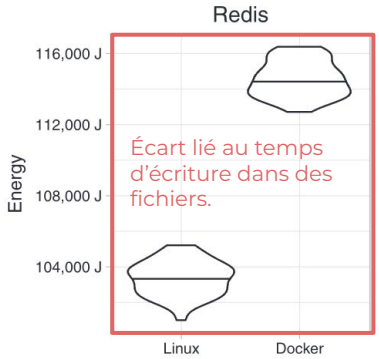


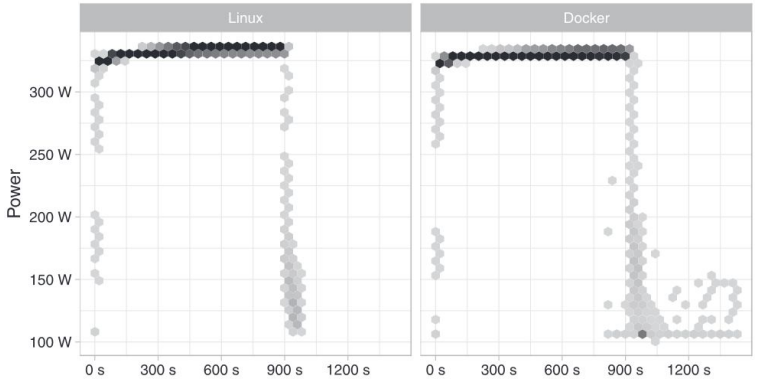
Case Study	Example Task	Linux	Docker	% Difference	$ Difference
Idle	One week idle	$1.20	$1.22	2.08%	+$0.02
WordPress				1.56%	
Redis	10^9 queries	$1.29	$1.43	10.85%	+$0.14
PostgreSQL	10^6 transactions	$0.55	$0.55	0.97%	+$0.00

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

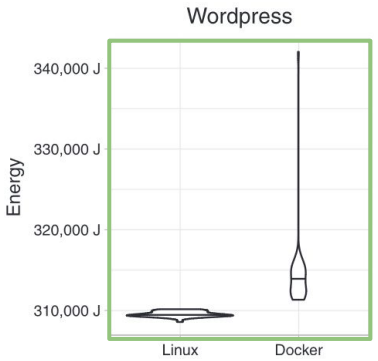
Utilisation de Docker et consommation d'énergie

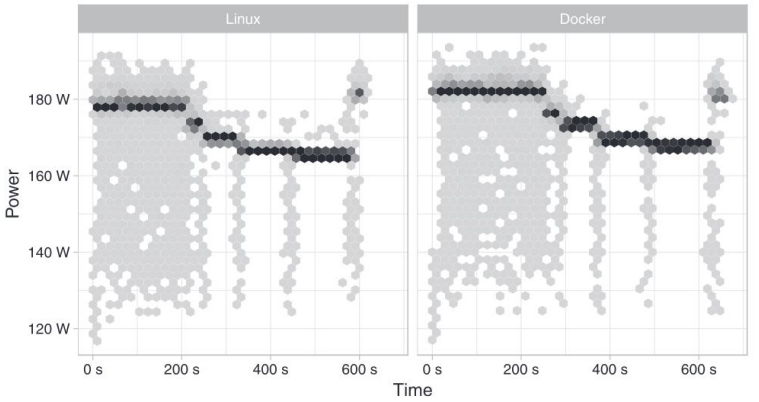






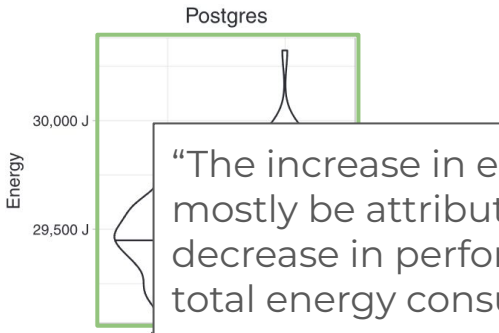
Case Study	Example Task	Linux	Docker	% Difference	$ Difference
Idle	One week idle	$1.20	$1.22	2.08%	+$0.02
WordPress				1.56%	
Redis	10^9 queries	$1.29	$1.43	10.85%	+$0.14
PostgreSQL	10^6 transactions	$0.55	$0.55	0.97%	+$0.00

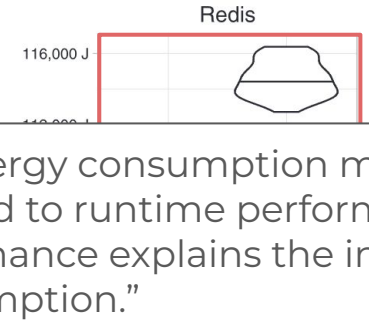


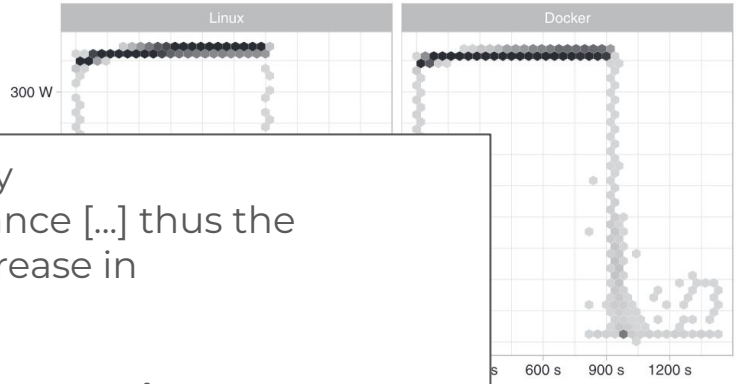


INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Utilisation de Docker et consommation d'énergie





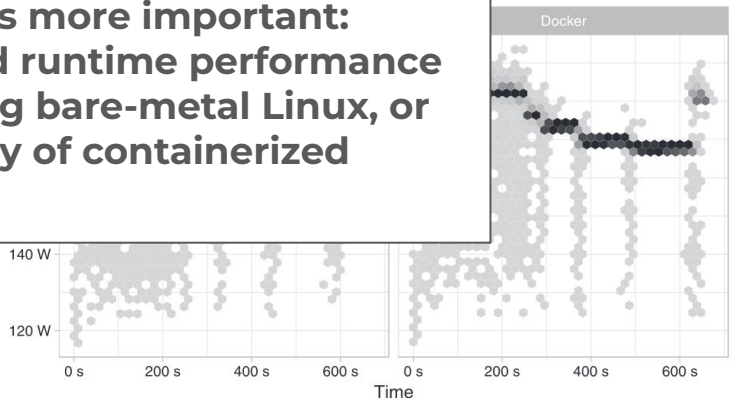


"The increase in energy consumption may mostly be attributed to runtime performance [...] thus the decrease in performance explains the increase in total energy consumption."

"Operations teams must decide which is more important: sustainability, energy consumption, and runtime performance of reduced resource usage by employing bare-metal Linux, or the process isolation and maintainability of containerized applications of Docker"

Case Study	Example Task	Lin			
Idle WordPress	One week idle	$1.			
Redis	10^9 queries	$1.			
PostgreSQL	10^6 transactions	$0.55	$0.55	0.97%	+$0.00





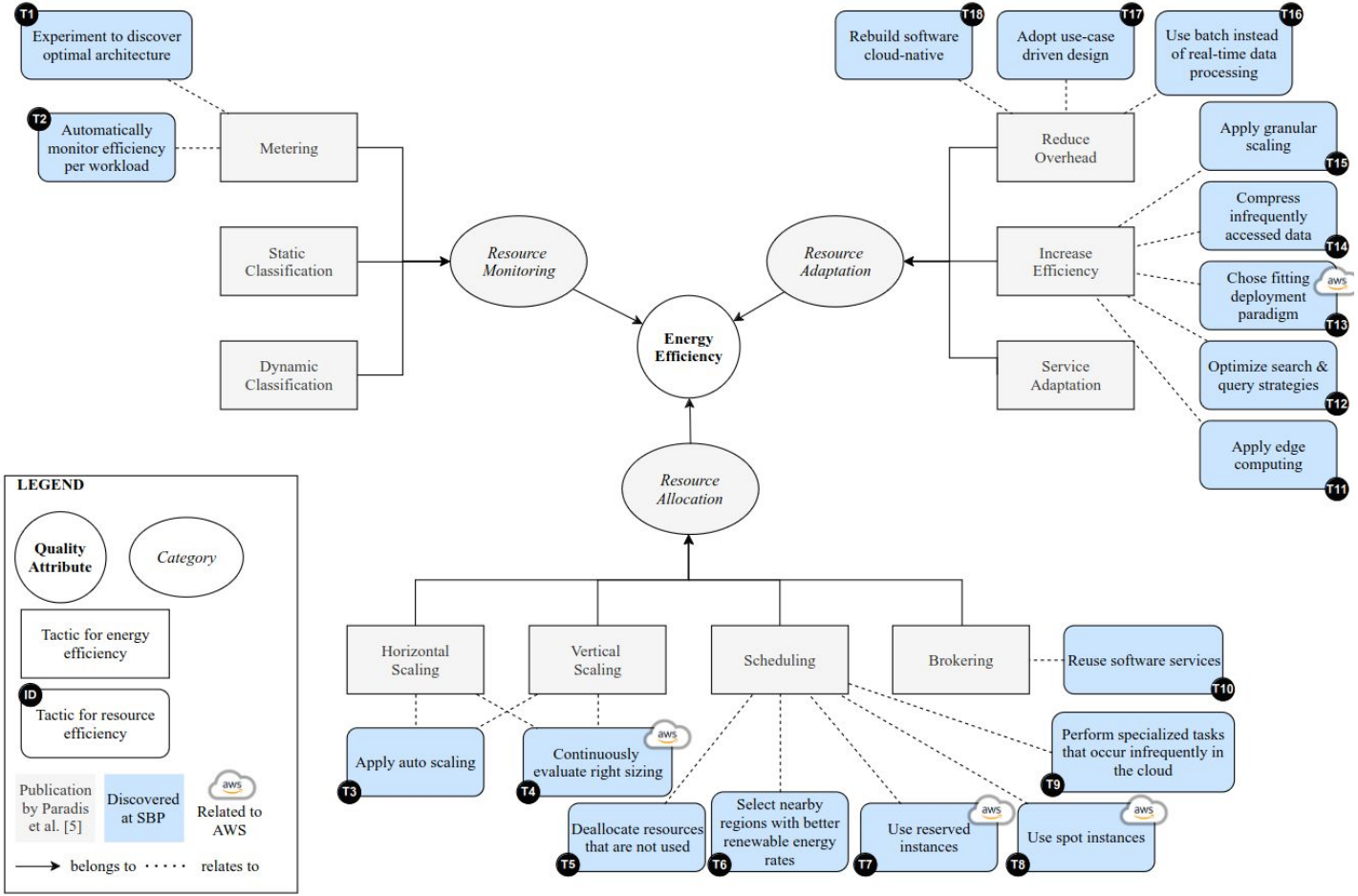
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Leviers (tactics) sur les architectures Cloud pour réduire la consommation d’énergie

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Leviers (tactics) sur les architectures Cloud pour réduire la consommation d'énergie

[Vos *et al.*, "Architectural Tactics to Optimize Software for Energy Efficiency in the Public Cloud", 2022]



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Conclusion

- La consommation d'un logiciel/service est fortement contextuelle (matériel, OS, threading, type d'usage, etc.)
- Nécessité de comparer et mesurer, car il y a des phénomènes contrintuitifs
- Plusieurs niveaux pour l'optimisation :
  - langages / pratiques d'implémentation
  - système / OS / Virtualisation (VM / Docker)
  - chaine de build / pipeline DevOps
  - Cloud et services
  - Bibliothèques utilisées
  - etc.