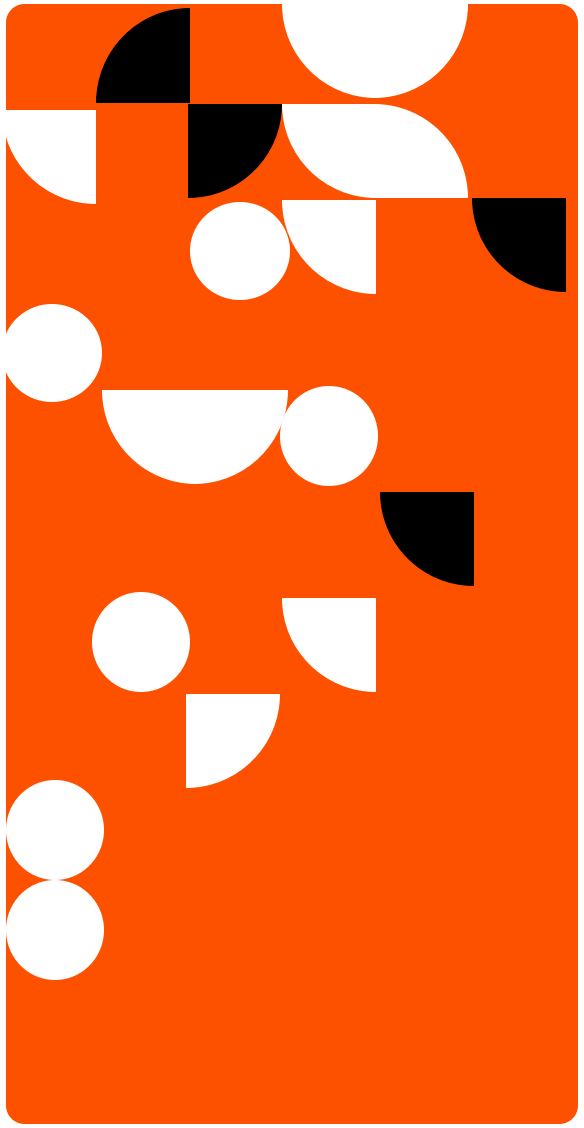
InterFLOP

Les travaux d'Aneo

20/03/2025

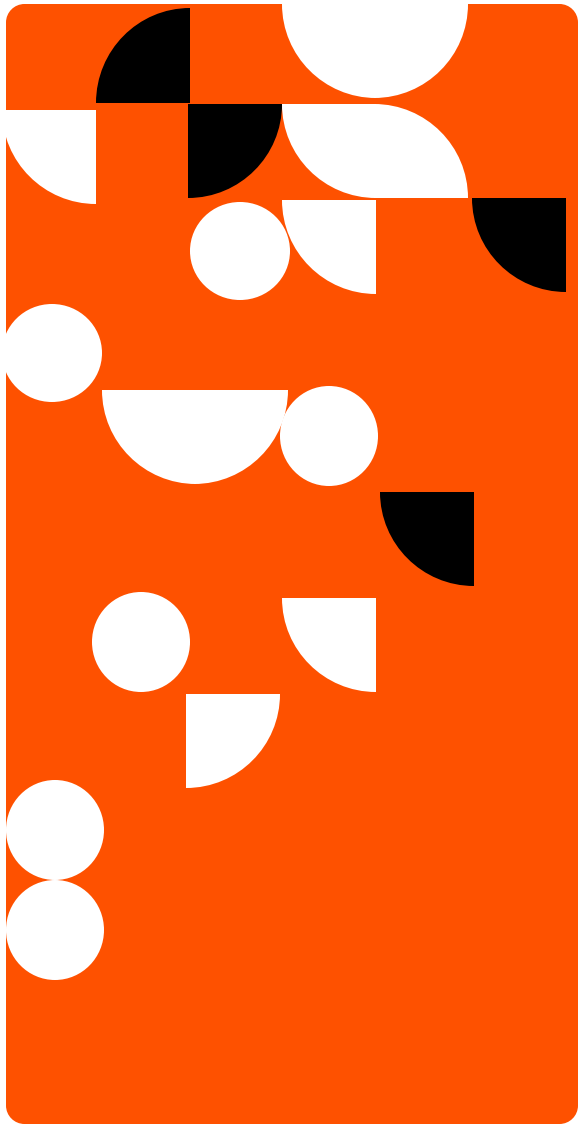
1. Introduction
2. PADLOC
3. PENE
4. Stratégie de test d’un backend
5. Vectorisation et enjeux d’API
6. Perspectives

1. Introduction



Objectifs

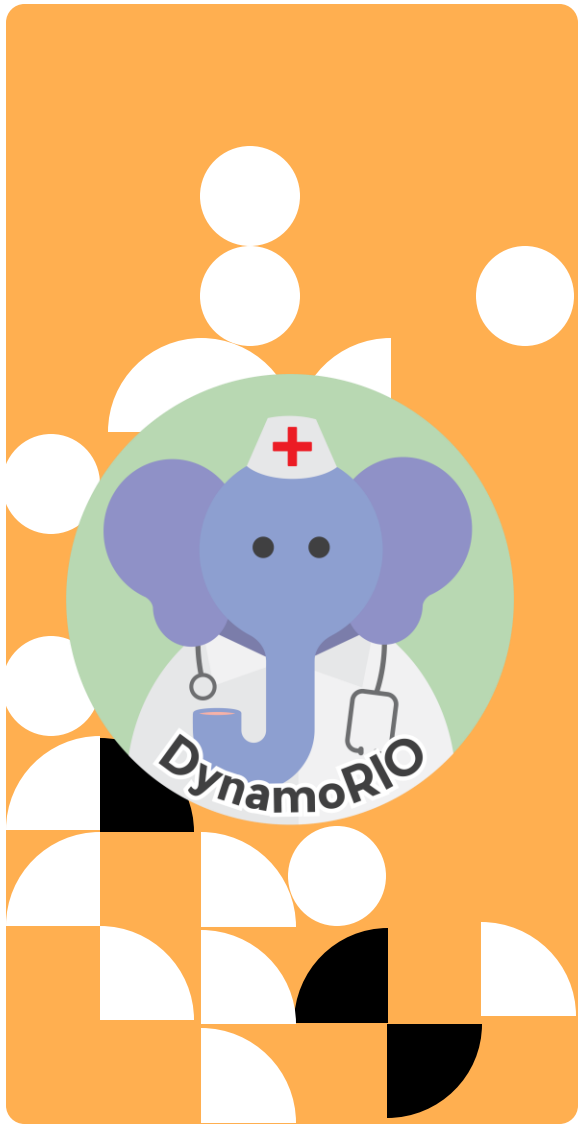
- Support ARM
- Support Windows
- Backends interchangeables
- Réduction du surcoût
- Interchangeable avec VERROU



Etat d'avancement

- Exploration d'autres solutions d'instrumentation
  - DynamoRIO
  - Creation de PENE
- Uniformisation de PENE/Verificarlo/Verrou
  - Generation de tests de backend
  - Stratégie de tests
  - Interface vectorielle

2. PADLOC



aneoconsulting/PADLOC

PADLOC

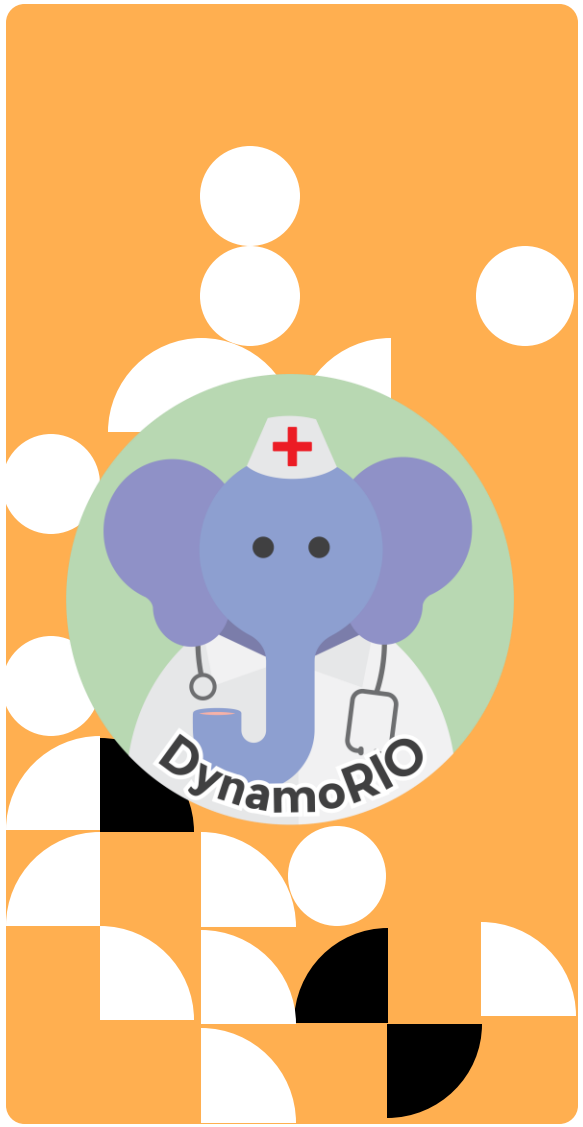
Un POC multi-plateformes avec DynamoRIO

Proficient **A**rithmetic **D**isturber for f**LO**ating point **C**omputation est un POC d'outil d'instrumentation binaire similaire à Verrou.

Réalisé par Dylan BRASSEUR, Mickaël TEAUDORS et Yoann VALERI lors d'un stage en été 2019.

Les objectifs :

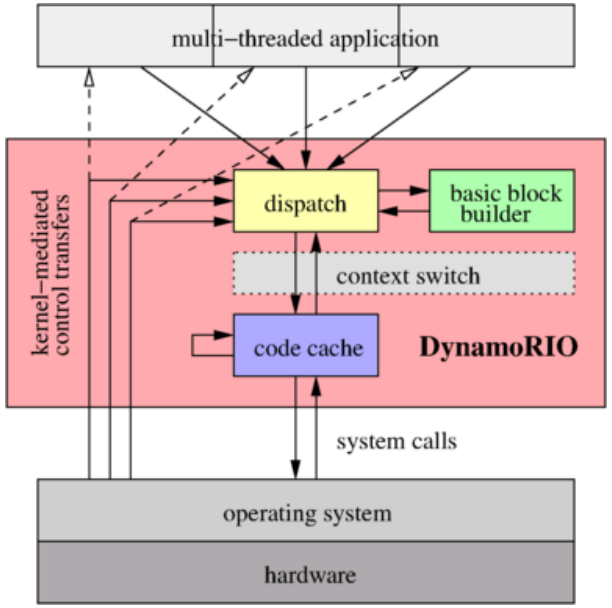
- Backend Verrou
- Windows
- x86-64 et AArch64



aneoconsulting/PADLOC

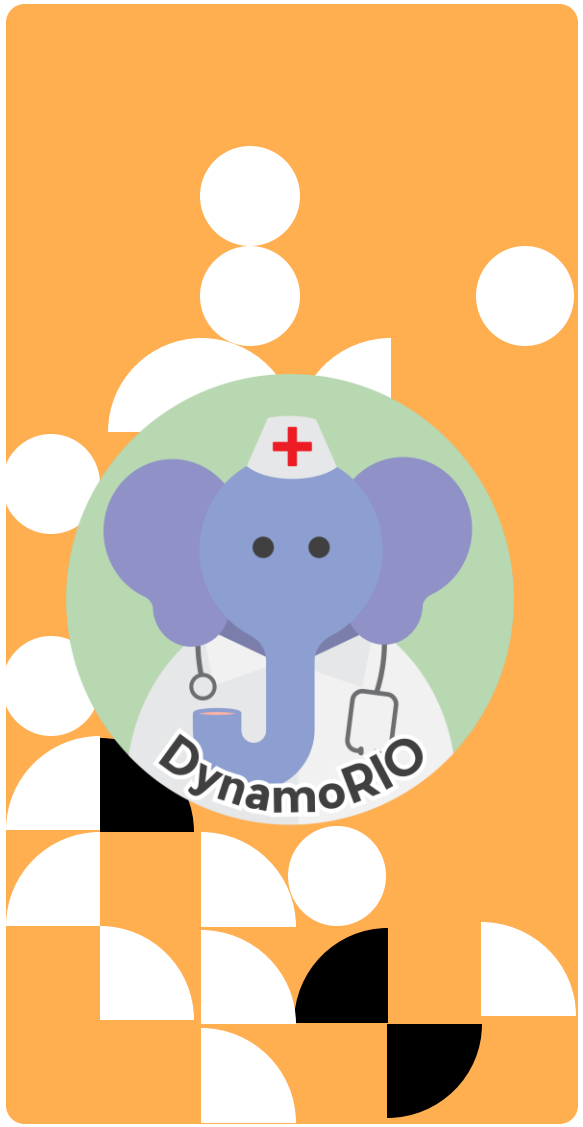
PADLOC

Comment fonctionne DynamoRIO ?



Avantages:

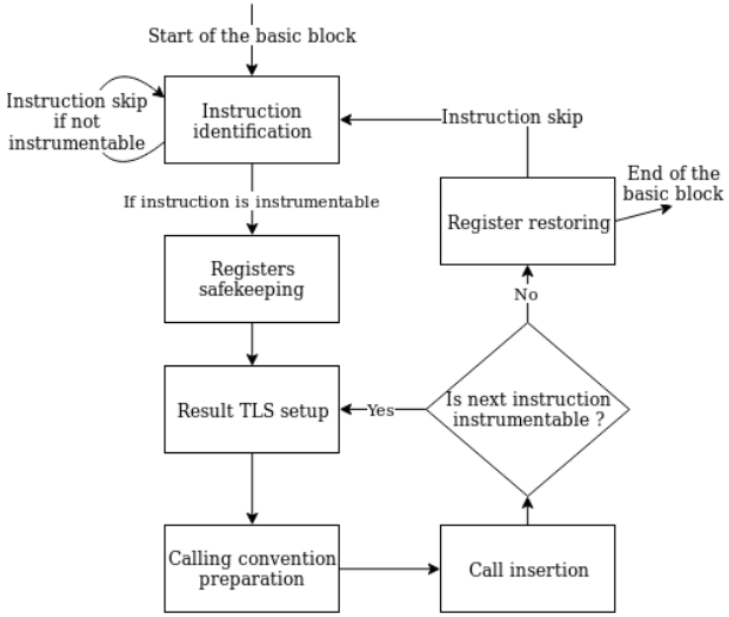
- Support d'application multi-threads
- Support de Linux et Windows
- Support de x86-64 et AArch64
- Instructions assembleur directement (pas d'IR)



aneoconsulting/PADLOC

PADLOC

Le résultat



Instrumentation de **SSE, AVX2, FMA**

Support de **Linux** et **Windows**

Backend Verrou implémenté

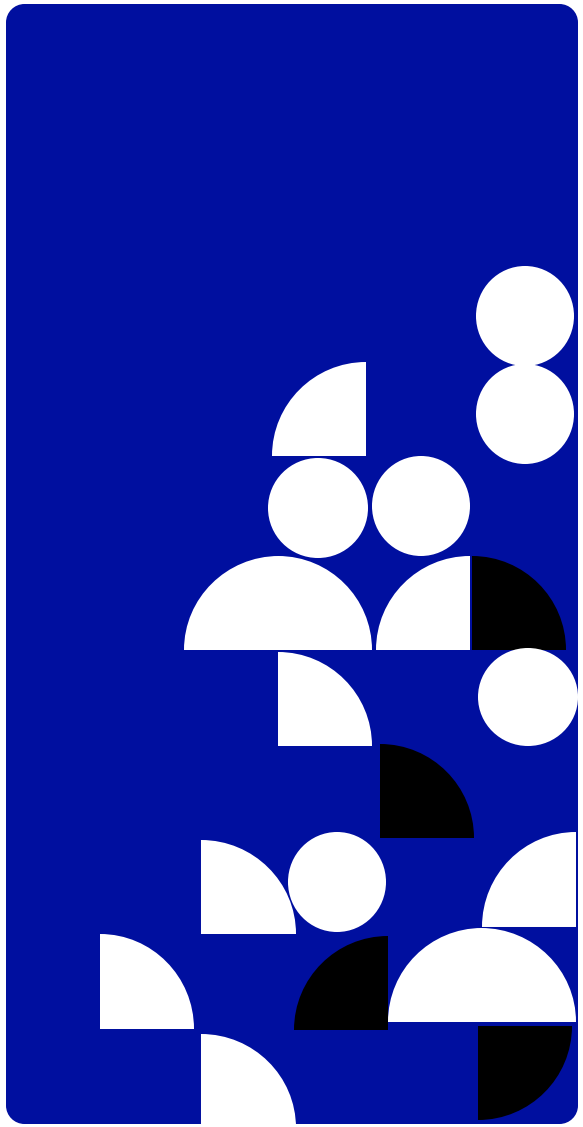
Support de **AArch64 non compatible** au moment du stage (Instructions non compatibles, problèmes de compilation, instructions manquantes ou incomplètes…)

Performances décevantes:

Program\Modification	*None*	*Verrou*	*Padloc*
Matmul 2000 * 2000	Real: 0.242s User: 1.426s	Real: 4m12.37s User: 4m12.37s	Real: 6m12.81s User: 46m31.38s
Pathtracer*	Real: 4.31s User: 4.31s	Real: 146.03s User: 146.03s	Real: 268.17s User: 268.17s

C'était en 2019, DynamoRIO a eu quelques MàJ depuis!

3. PENE



aneoconsulting/PENE

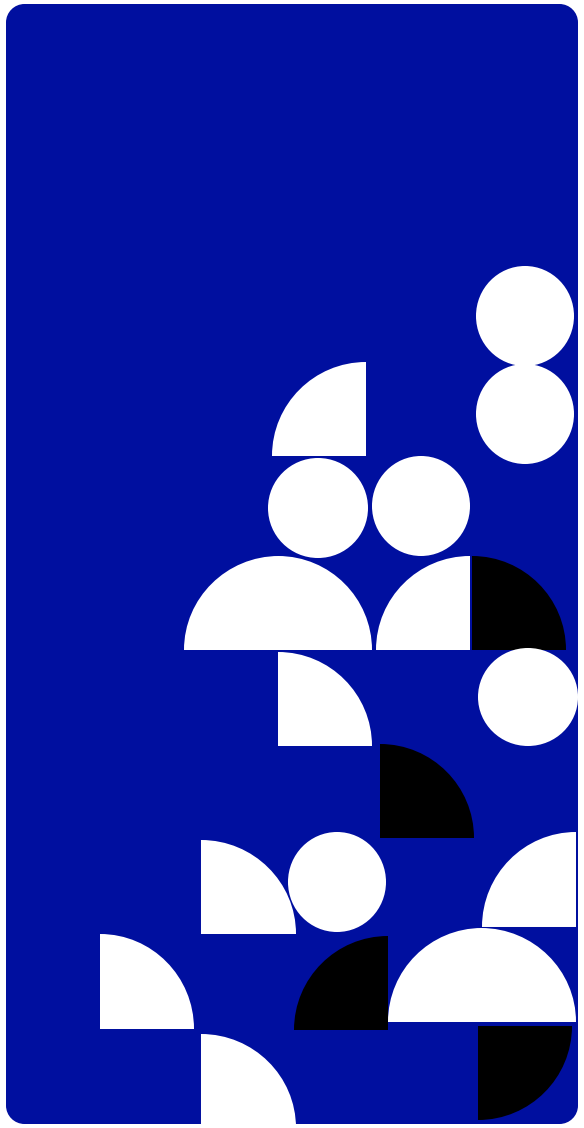
PENE

Outil d'instrumentation dynamique des instructions flottantes avec Intel Pin

Pin **E**nabled **N**umerical **E**xploration est un outil d'analyse d'erreurs flottantes, multi-plateforme et optimisé pour les plateformes x86.

Les objectifs :

- Optimisation/réduction du temps d'instrumentation
- Compatibilité Windows et Linux
- Support des architectures x86-64 et de toutes les variations d'instructions SIMD
- Faciliter l'ajout de backend



aneoconsulting/PENE

PENE

Outil d'instrumentation dynamique des instructions flottantes avec Intel Pin

Avantages :

- Compatibilité Windows et Linux
- Permet l'accès aux registres vectoriels
- Permet l'écriture d'analyses avec une granularité instruction
- Supporte l'instrumentation d'applications multi-threads

Inconvénients :

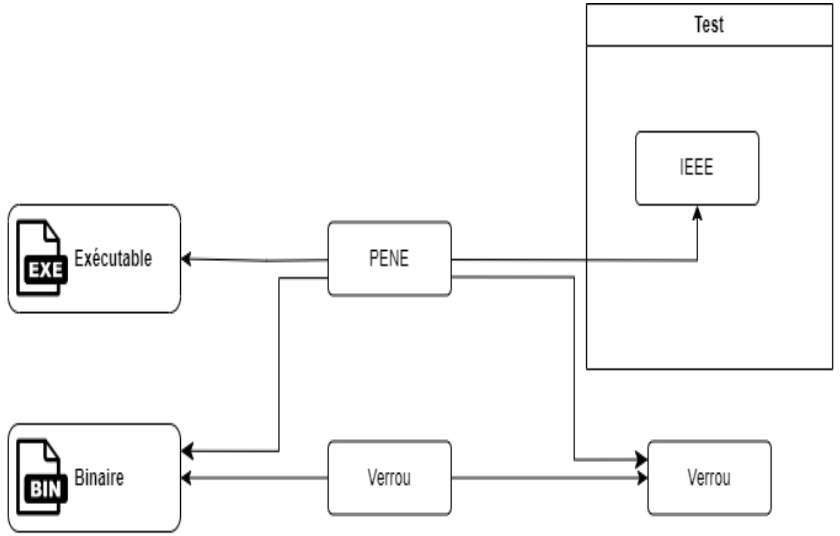
- Compatible uniquement sur les plateforme x86
- Opère au niveau des instructions assembleurs (pas d'IR)

PENE

aneoconsulting/PENE

Fonctionnement de PENE

- Exécution du programme
- Interception des opérations flottantes au runtime
- Appel à des backends
- Dévectorise les instructions SIMD

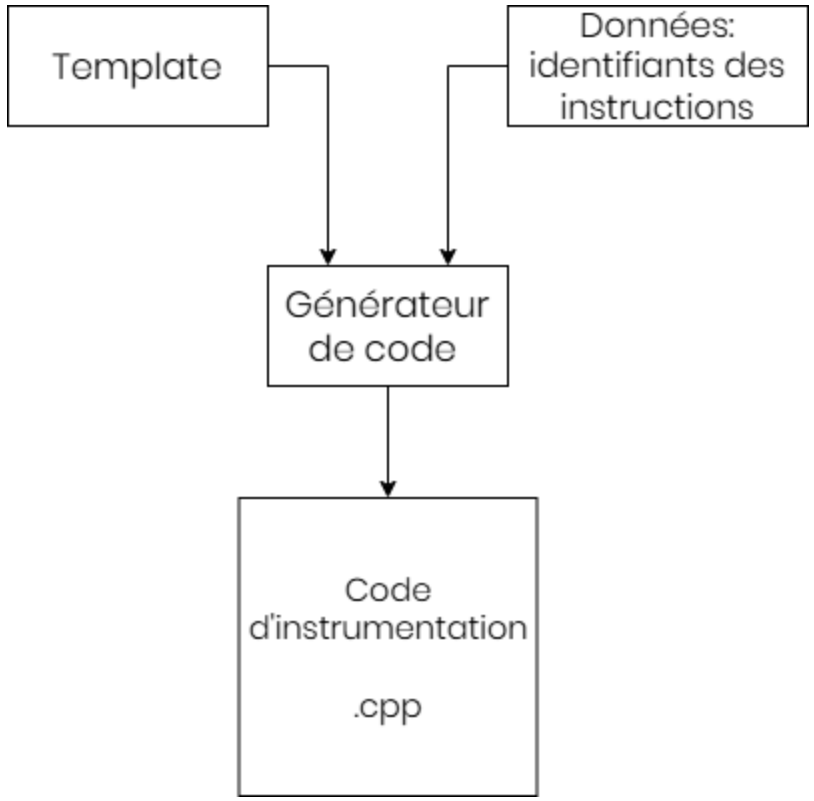


aneoconsulting/PENE

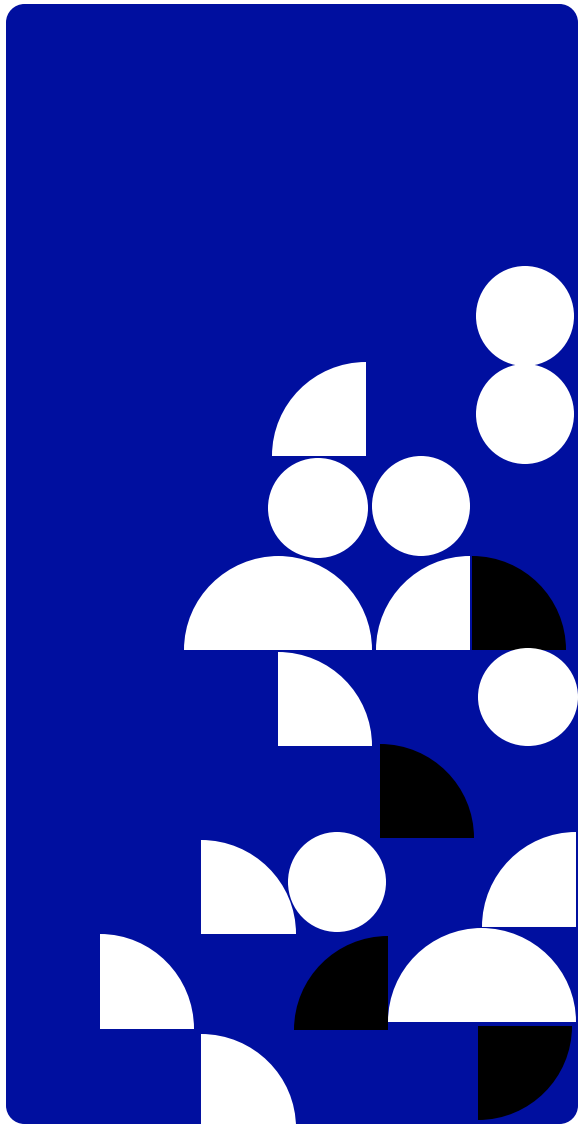
PENE

Générateur de code

- Instrumentation à la granularité d'une instruction
- => Plus de liberté pour l'instrumentation, accès au type d'instrumentation, registres, etc..
- => toutes les variantes d'instruction à gérer
- => nécessité d'un générateur de code pour l'instrumentation



600 variantes d'instructions supportées grâce à la génération de code pour les jeux d'instruction **SSE**, **AVX**, **AVX512** et **FMA**



aneoconsulting/PENE

PENE

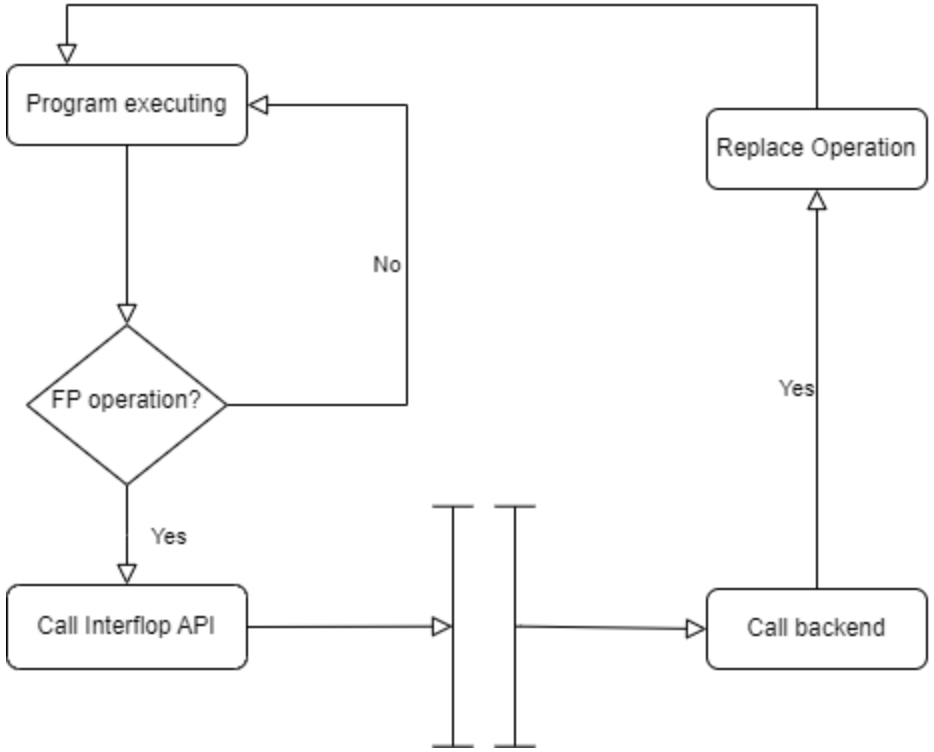
Exclusion des parties du code de l'instrumentation

- Intel Pin offre la possibilité de choisir les blocs de code à instrumenter
- PENE permet de définir une liste de fonctions à exclure
- Première étape pour l'algorithme de Delta Debug

4. Stratégie de test d'un backend

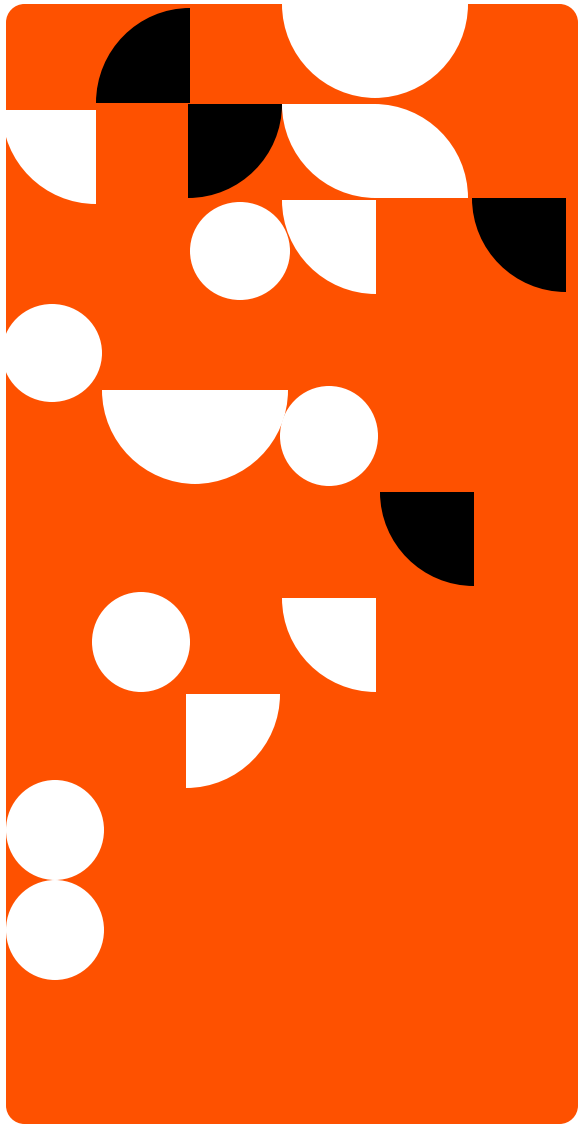
Implémentation de l'interface InterFLOP dans PENE

Une première étape pour uniformiser la notion de backend



- Implémentation de l'interface InterFLOP
- Réutilisation de la librairie standard InterFLOP
- Branchement dynamique de backends

aneoconsulting/interflop-backend-adapter



Génération de code

Automatisation de la génération de backend

- Génération du code des backends
- Génération des tests unitaires des backends
- Compilation en fonction du frontend cible

Exemple de backend simple :

```
add_float:
    *cptr = a+b;
end_add_float
sub_float:
    *cptr = a-b;
end_madd_float
...
```

aneoconsulting/interflop-backend-adapter

Stratégie de test

Test de l'instrumentation

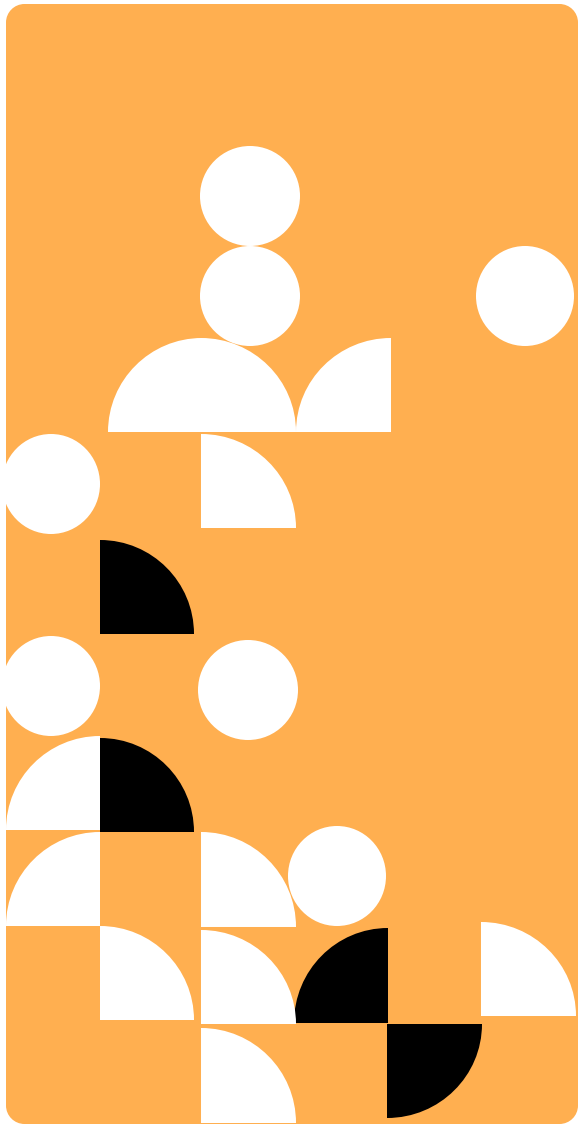
- Tests unitaires existants
- Ajout de tests d'intégration (vérification de l'instrumentation)

Proposition :

- Mise en commun des tests unitaires + traduction en templates
- Backend modifiant les bits de poids fort du résultat

aneoconsulting/interflop-backend-adapter

5. Vectorisation et enjeux d'API



Interface vectorielle

Un POC d'interface supportant le SIMD

a-hamitouche/interflop-backend-verrou

- Passage des valeurs par pointeur
- Utilisation d'intrinsèques
- Métaprogrammation/compilation multi-architecture

Enjeux :

- Minimiser la perte de performances :
  - Minimiser la perte durant l'exécution
  - Minimiser la perte due à une différence de compilateur frontend/backend

Interface vectorielle

Un POC d'interface supportant le SIMD

a-hamitouche/interflop-backend-verrou

Approche:

- Portage vectoriel de certains backends de Verrou (SSE, AVX)
- Instrumentation d'un dotprod écrit en intrinsics avec des tailles de vecteurs croissantes

Résultats :

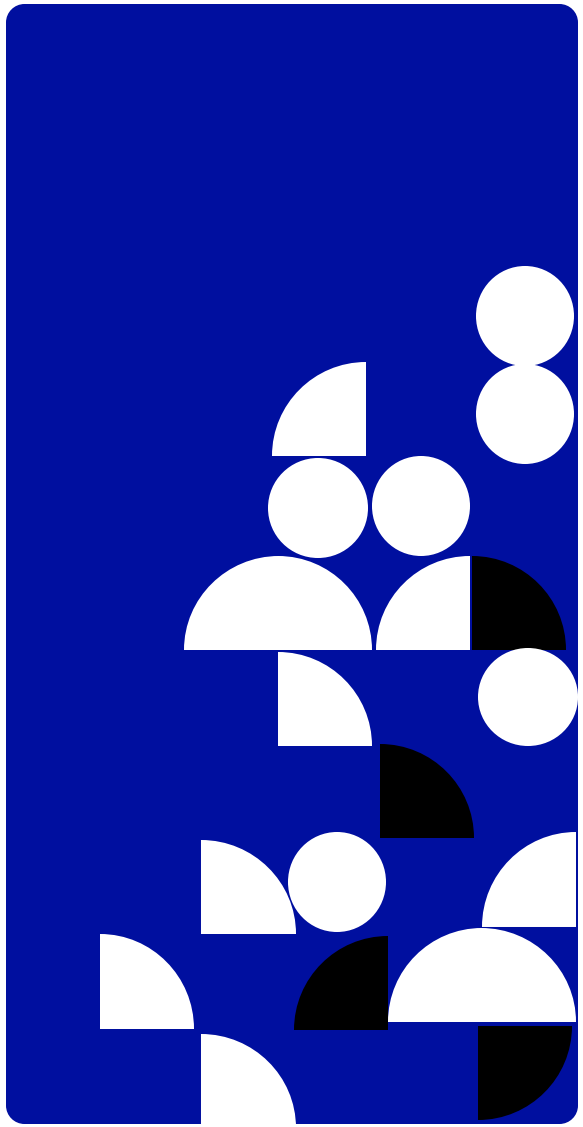
	Scalaire	Vectoriel (AVX)
128	7033	6884
256	6547	2770
512	10552	5293
1024	21438	10384
2048	41404	20371
4096	79280	106660
8192	152322	91666

Comparaison du temps d'exécution en cycles du mode d'arrondi upward scalaire/vectoriel sur un dotprod

	Scalaire	Vectoriel (AVX)
128	7031	4719
256	5331	2852
512	10324	5781
1024	20388	11433
2048	46053	22743
4096	87728	47647
8192	185457	97009

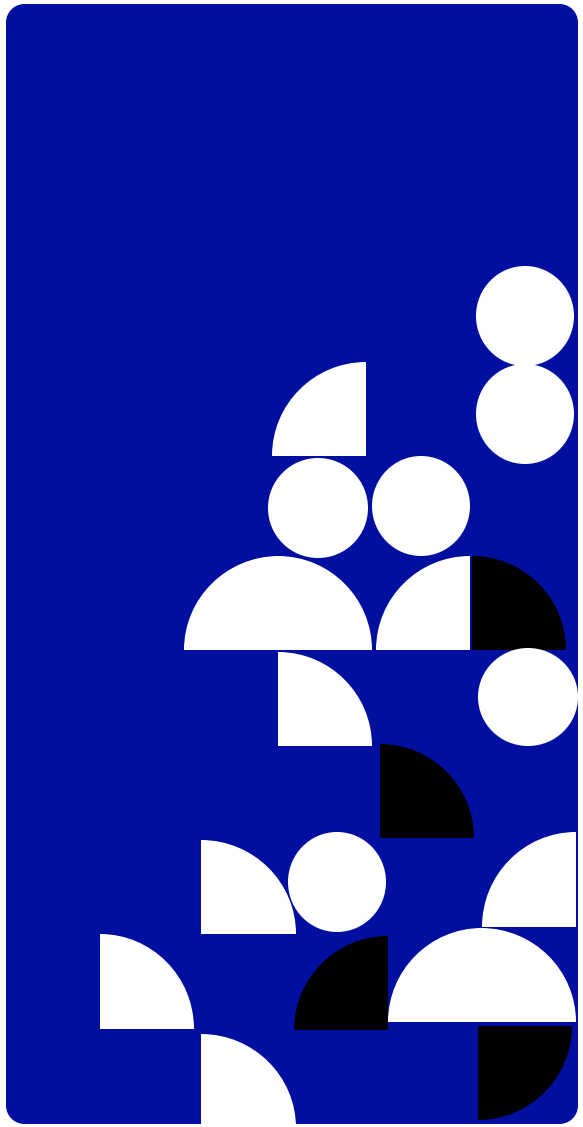
Comparaison du temps d'exécution en cycles du mode d'arrondi downward scalaire/vectoriel sur un dotprod

6. Perspectives



Bilan

- ❌ Support ARM
- ✅ Support Windows
- ✅ Backends interchangeables
- ✅ Réduction du surcoût
- 🚧 Interchangeable avec VERROU



Suite des travaux

- Intégration des interfaces vectorielles dans les différents outils
- Delta debug dans PENE

FPT4

- Parallélisation du delta debug avec ArmoniK

Vos contacts

Wilfried

CTO and Head of R&D Advanced Computing Technologies

wkirschenmann@aneo.fr

+33 6 85 44 67 73

Jérôme Gurhem

ArmoniK Tech Lead

jgurhem@aneo.fr

Florian Lemaitre

Expert Cloud HPC

flemaitre@aneo.fr

+33 6 32 99 50 75

Dylan Brasseur

Consultant Senior

dbrasseur@aneo.fr

+33 6 58 69 21 70

122 Avenue du Général Leclerc

92100 Boulogne-Billancourt

01 46 44 24 66

www.aneo.eu