

Étude, implémentation et optimisation de compilateurs

Compilateurs d'inférence IA TVM · MLIR · StableHLO

Zhangshanying | ZY2424104

Beijing Sulu Technology Co., Ltd.

Beijing, Chine · Chaîne d'outils logiciels

20 avril – 20 octobre 2026

Parcours : contexte → TVM → MLIR → bilan et perspectives

Une chaîne de compilation au service du produit

Produit

Transcription assistée
Contextes judiciaires
Usages en continu et hors ligne

Organisation

Algorithmes : modèles
Chaîne d'outils : déploiement
Système : intégration produit

Mon rôle

Importation des modèles
Optimisation des graphes
Interfaces entre composants

Modèles



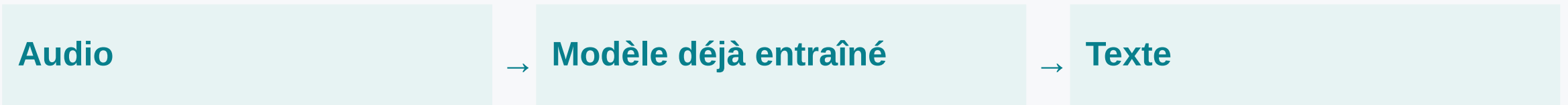
Compilateur



Puces chinoises / moteur privé

Du modèle entraîné au programme qui fonctionne

Inférence : utiliser un modèle déjà entraîné pour produire un résultat.



Le compilateur prépare les calculs pour le matériel choisi.



Mon rôle : organiser ces calculs, leur mémoire et leur validation.

Exemple de transcription simplifié · Repères : documentation TVM et MLIR.

Lire un graphe : une couche de réseau de neurones

Exemple courant : $Y = \text{ReLU}(XW + b)$

1 · Matmul

Combiner les données avec les poids appris.

→

2 · Ajouter un biais

Ajouter un décalage appris par le modèle.

→

3 · ReLU

Remplacer les valeurs négatives par zéro.

Exécution séparée : des tableaux intermédiaires sont écrits puis relus.

Fusion possible : Matmul + biais + ReLU dans un même calcul

Objectif : moins de transferts mémoire, avec un résultat numérique vérifié.

Exemple pédagogique, selon les capacités du matériel · Source : Apache TVM, Operator Fusion.

Pourquoi faire évoluer la chaîne ?

Mémoire élevée

Les essais de plusieurs modèles ou versions sont limités.

Objectif : réduire les allocations inutiles.

Cycles de test longs

Débogage et tests prennent du temps.

Objectif : faciliter les comparaisons et le diagnostic.

Nouveaux matériels

La voie Relay demande des adaptations coûteuses.

Objectif : organiser une base MLIR plus facile à faire évoluer.

Choix fondés sur le profilage et l'étude des cadres et publications.

Deux axes et une voie de validation

Axe 1 · Architecture existante



Axe 2 · Nouvelle architecture



Validation numérique : sorties TVM ↔ sorties du simulateur

Schéma de synthèse · TP = parallélisme de tenseurs

Jalons : réalisé, en cours et prévu

Prise en main et diagnostic TVM	TERMINÉ
Correction mémoire à l'importation	INTÉGRÉE
Validation TVM—simulateur	RÉALISÉE
Interface egg et cadre fusion / TP	RÉALISÉS
Adaptation de l'IR de graphe MLIR	EN COURS
Prise en charge des modèles quantifiés	PRÉVUE

Environ 3 mois : outils et TVM · Environ 3 mois : MLIR

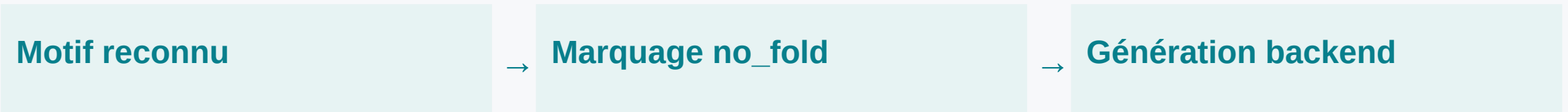
Éviter la matérialisation du masque

Masque d'attention : indique quelles positions peuvent être utilisées.

Avant : un grand tenseur est créé pendant l'importation



Après : le backend génère le masque demandé

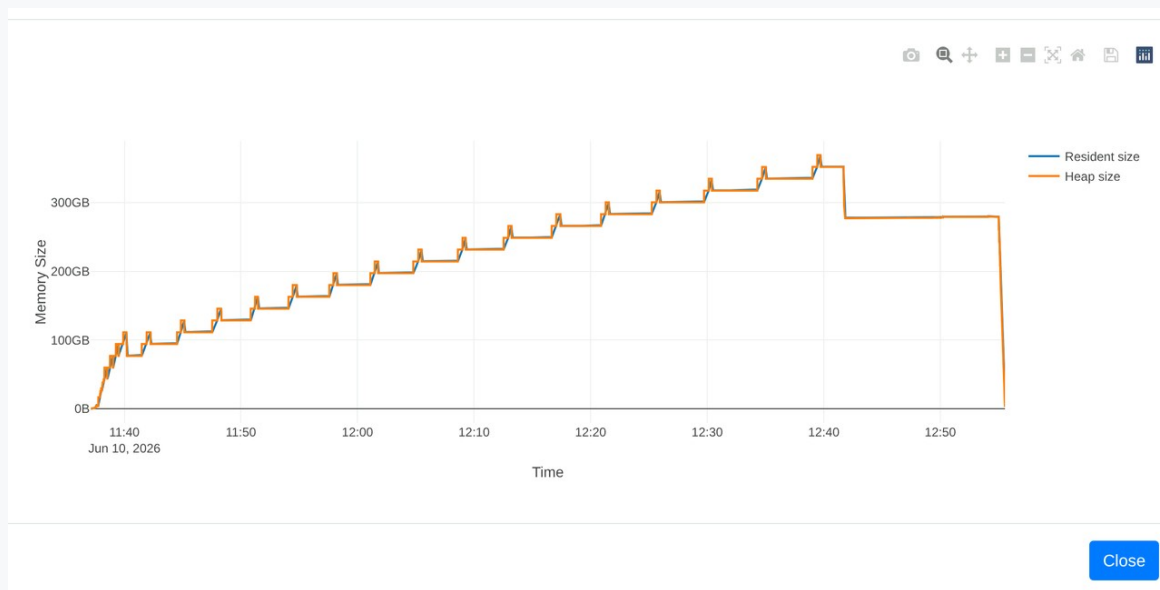


Configurations concernées : séquences jusqu'à 4k ou 128k.

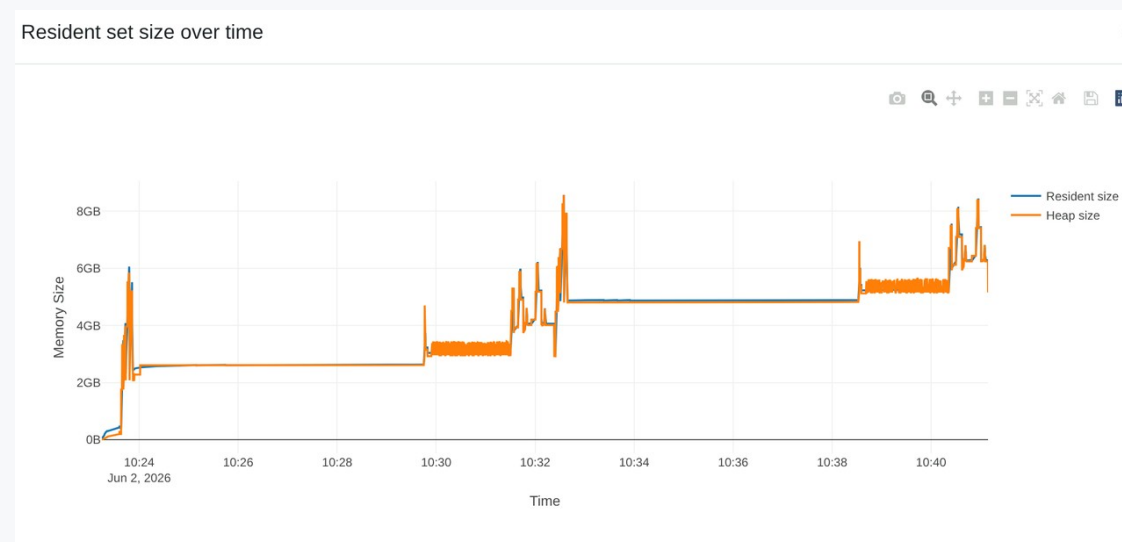
no_fold est un mécanisme du projet, pas un attribut standard ONNX.

Les courbes illustrent l'effet sur la mémoire

Avant · cas à plusieurs couches



Après optimisation du masque



Même modèle · Illustration qualitative · Échelles verticales différentes

Une correction intégrée et un contrat à adapter

opt: default nofold to matched attention masks

Merged Shanying Zhang requested to merge `opt/no_fold` into `main` 3 months ago

Overview 0 Commits 2 Pipelines 0 Changes 3

- add default nofold to matched attention masks

this can be disabled by pass `skip_attention_mask_chain_fold: False` to `relay.frontend.from_onnx`'s `convert_config`

- port minor opt

0 0

8✓ Approval is optional

Merged by Jiangsha Ma 2 months ago [Revert](#) [Cherry-pick](#)

Merge details

- Changes merged into main with [40743d20](#).
- Deleted the source branch.

Activity All activity

- Shanying Zhang requested review from [@jiangsha](#) 3 months ago
- Shanying Zhang assigned to [@zhangshanying](#) 3 months ago
- Shanying Zhang added 2 commits 3 months ago
 - [d5d68cc2](#) - feat: add attention mask match and skip fold const
 - [43da8c6a](#) - opt: Disable InferType if it was done and no changes after previous pass[Compare with previous version](#)
- Jiangsha Ma mentioned in commit [40743d20](#) 2 months ago
- Jiangsha Ma merged 2 months ago

Ma contribution

Reconnaître le masque

Modifier l'importeur

Livrer la correction

Coordination des passes

Ancien matcher : masque instancié

Nouvelle représentation : adaptation
avec les collègues responsables

Vérifier les résultats avec TVM et le simulateur



Comparaison numérique

Cosine · L2 · Erreur absolue · Erreur relative

Diagnostic

Activer / désactiver
les passes concernées

Le dispositif est réalisé ; aucun seuil ou taux de réussite non fourni n'est annoncé.

Un langage commun pour préparer les calculs



IR : description du programme

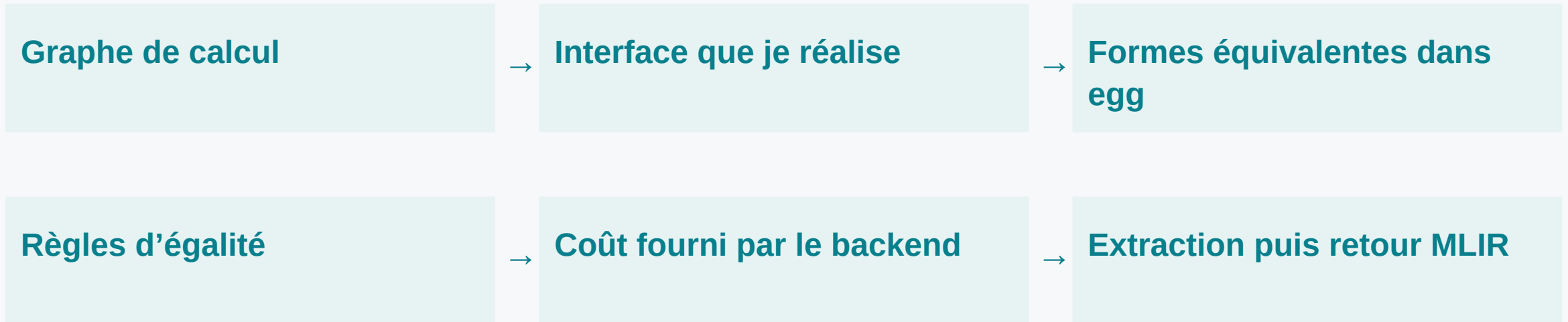
StableHLO décrit les calculs du graphe.
MLIR fournit le cadre de transformation.
Le backend prépare la cible matérielle.

État actuel

Adaptation de l'IR : en cours
Interfaces et cadre : réalisés
Modèles quantifiés : prochaine étape

Sources : MLIR Language Reference [R1] · StableHLO specification [R2].

Egg compare plusieurs façons de calculer



Intégration

Option déjà disponible dans la chaîne

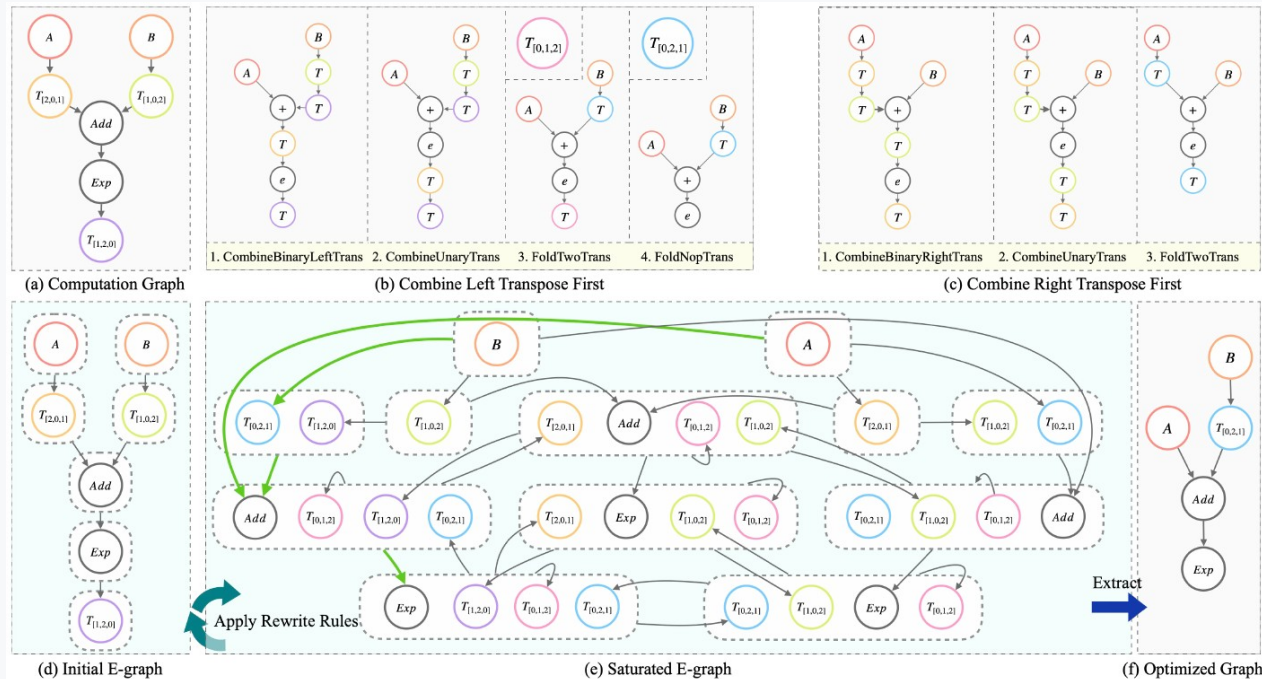
Limite

Modèle de coût encore à améliorer

Sources : egg [R3] · Exemple public d'interface [R8] (egglog, distinct du projet).

Un exemple publié pour comprendre les réécritures

Transpose : changer l'ordre des dimensions d'un tableau.



Déplacer des transpositions peut rendre d'autres optimisations possibles.

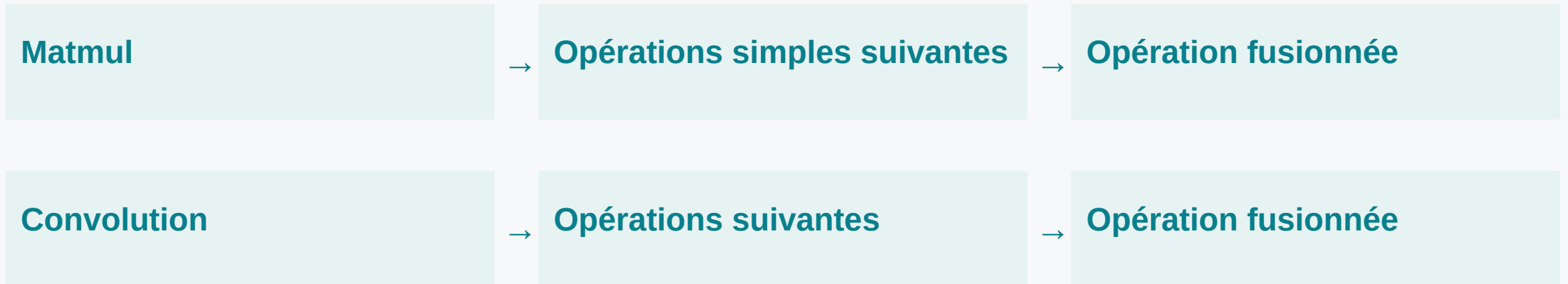
La validité dépend des opérations, des formes et des permutations.

Hui Guo et al., nncase, 2025 · Figure 2, p. 5 · arXiv:2512.21571v1 · CC BY-SA 4.0.

Extrait de la figure publiée, rendu PNG ; illustration, pas un résultat personnel.

Migrer les motifs de fusion vers MLIR

Exemples de motifs du périmètre



Autres opérateurs : bibliothèque fermée, selon les possibilités du matériel.

Mon travail principal : organiser les règles et les interfaces du cadre.

Schémas fonctionnels ; pas de capture d'un résultat de fusion particulier.

Répartir les calculs avec des règles communes

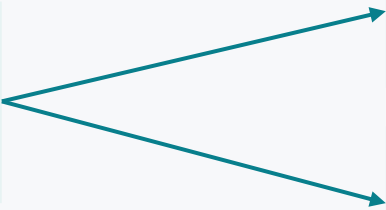
Interface = contrat commun

Ce contrat précise comment un opérateur peut être fusionné ou réparti.
Chaque opérateur expose ses contraintes.

TP = travail réparti

Des parties d'un tenseur sont réparties entre plusieurs unités de calcul.
Shardy propage les règles de répartition.

Tenseur global



```
graph LR; A[Tenseur global] --> B[Partie 1 - unité A]; A --> C[Partie 2 - unité B];
```

Partie 1 · unité A

Partie 2 · unité B

Répartition simplifiée ; le découpage dépend des opérations et de leurs règles.

Fusion : regrouper des opérations · TP : répartir des tenseurs · Tiling : blocs locaux.

Des risques situés aux frontières des composants

Masque / Flash Attention

Risque : hypothèse du matcher différente

Suivi : coordination importeur–passes

MLIR / egg

Risque : types ou formes mal reconstruits

Contrôle : cas simples et vérification IR

Coût d'extraction

Risque : forme moins adaptée au matériel

Suite : améliorer le coût backend

Modèles quantifiés

Risque : types / paramètres incohérents

Prévu : comparaison TVM–simulateur

Mes contributions sont des mécanismes livrés

Contributions personnelles

TVM : reconnaissance du masque
et modification de l'importation

Egg : interface optionnelle

Fusion / TP : cadre et contrats

Validation : interface TVM–simulateur

Open	3	Merged	8	Closed	2	All	13	Bulk edit	New merge request	⋮
Recent searches	▼	Author	=	Shanying Zhang	×	Updated date	▼	⌵		
clean(passes): clean up unused passes										
1813 · created 1 month ago by Shanying Zhang										
chore ci-passed										
feat(tvm_pass): support dump tvm ops										
1807 · created 1 month ago by Shanying Zhang										
ci-passed										
release tvm pass regression										
1773 · created 2 months ago by Shanying Zhang										
ci-failed feat										
refactor(passes): make opt_pass_seqs as module-level										
1786 · created 2 months ago by Shanying Zhang										
ci-passed refactor										
fix(sam2): use parsed case_num in sort_data_by_index										
1763 · created 2 months ago by Shanying Zhang										
fix tested										
fix: hwptn in fuse_silu_quant										
1761 · created 2 months ago by Shanying Zhang										
fix tested										
opt: reduce memory usage during compilation										
1717 · created 3 months ago by Shanying Zhang										
Merged										
chore: optimize calcc_log func, disable calcc_log ANSI colors on redirected output										
1700 · created 4 months ago by Shanying Zhang										
Merged										

Trace de modifications intégrées

Statut de fusion ≠ validation de tous les tests

Compétences développées et projet professionnel

C2C1 · Modéliser

Distinguer expression
et tenseur matérialisé.

Relier MLIR et egg
avec des contrats clairs.

C2C2 · Arbitrer

Relier profilage,
choix techniques
et limites.

Vérifier les sorties.

C4C1 · Progresser

Anticiper les effets
entre les modules.

Mieux expliciter
les conventions.

Poursuivre en compilateurs IA : algorithmes + logiciel + architecture matérielle.

Réduire les coûts et préparer la suite

Ce qui est livré

Correction mémoire TVM intégrée
Interface egg optionnelle
Cadre de fusion et de TP
Validation TVM—simulateur

Ce qui continue

Adaptation de l'IR MLIR : en cours
Modèles quantifiés : prévus
Coût backend : à améliorer
Contrats et couverture : à renforcer

Merci pour votre attention. Je suis prêt à répondre à vos questions.