Preferred Networks 2022年度インターン選考 テーマ別課題

JP01. MN-Core向けのコンパイラ及び周辺ライブラリの開発

課題について

想定所要時間は最大1日である。全課題が解けていなくても提出して構わない。

提出物

以下をzipファイルにまとめて提出してください。

- 課題1のレポート
- 課題2のレポート
- 課題3のプログラムのソースコード
- 課題3の添付のONNXに対する出力
- 課題3のプログラムとその出力の簡単な説明

課題1

深層学習ワークロードは事前に計算グラフが確定するなどの特性から、従来の一般的なプロセッサー向けのコンパイラ以上の最適化が行える。深層学習向けコンパイラにおいて行われる典型的な最適化操作についていくつか列挙し、その操作や効果について簡単にまとめよ。

課題2

DNNフレームワークのPyTorchは、計算グラフ中の複数のオペレータを融合して効率化を図る最適化を実装している。この最適化はfusionと呼ばれる。fusionの具体例を調べ、それが計算速度、メモリ使用量にどのような影響を与えるか定量的に述べよ。

ヒント
https://github.com/pytorch/pytorch/blob/45a7ae19293fad24bdacc93dc8ff93521ad73c2c/torch/csrc/jit/python/init.cpp

課題3

Relu, Log, Negなどの単項オペレータが連続しているときは、それらを融合することで高速化できることが知られている。添付のONNX(strange_resnet18.onnx)について、このfusionが適用可能なオペレータの集合を出力せよ。また、余力があれば Add を融合する場合の集合も出力せよ。出力の形式は自由に定義してよいが、必要であれば以下の例を参考にせよ。

出力例

```
['Relu(input.24) -> (onnx::Log_198)', 'Log(onnx::Log_198) -> (onnx::Neg_199)',
'Neg(onnx::Neg_199) -> (input.28)']
['Relu(input.56) -> (onnx::Log_228)', 'Log(onnx::Log_228) -> (onnx::Neg_229)',
'Neg(onnx::Neg_229) -> (input.60)']
```

ヒント
strange_resnet18.onnxをパースするプログラム。解答にはこのプログラムを使わなくても良い。

```
import collections
import sys

import onnx


def node_str(node):
    return "{}({}) -> ({})".format(node.op_type, ", ".join(node.input), ",
".join(node.output))


def create_value_to_users(model):
    value_to_users = collections.defaultdict(lambda: [])
    for node in model.graph.node:
        for input in node.input:
            value_to_users[input].append(node)

    return value_to_users


def main():
    model = onnx.load("strange_resnet18.onnx")
    value_to_users = create_value_to_users(model)
    for value, users in value_to_users.items():
        users_str = "["
        for u in users:
            users_str += node_str(u) + ", "
        users_str += "]"
        print(value + ": " + users_str)


if __name__ == "__main__":
    main()
```