Vector Pack in VeGen

1 What's in a Vector Pack

A vector pack is not a collection of vectors, but rather a collection of instructions, each of which computes a scalar value.

2 Kinds of Vector Packs

2.1 General

To create a general vector pack, we need a `VectorPackContext`, an array of `Matches`, two bit vectors of elements and their dependencies, an intrinsic that is the producer of this pack, and the LLVM `TargetTransformInfo`.

2.2 Phi

To create a Phi pack, we do not need `Matches` as we do for creating a general pack.

2.3 Load

To create a Load pack, we do not need `Matches` since we already set it to be the load operation. Additionally, we need a condition pack to prevent unwanted reads from improper addresses and a flag to handle non-consecutive loads.

2.4 Store

Same as Load.

2.5 Reduction

2.6 GEP

GEP stands for "get element pointer".

2.7 Gamma

A Gamma pack is also called a Gated Phi Pack. It is a Phi node with incoming blocks replaced with explicit control conditions.

2.8 Cmp

3 Construction of Vector Packs

To construct a vector pack, we need to perform three steps in common, in addition to filling the vector pack context.

3.1 computeOperandPacks

This step has two sub-steps: compute and canonicalize. The compute sub-step gathers matched values into an array, mainly using a structure named OperandPack. This structure stores the vector type and the producers of the operand. The canonicalize sub-step wraps the OperandPack with a unique pointer and uses a map to guarantee uniqueness.

3.2 computeOrderedValues

For a general pack, it checks the `Matches` and filters out the unmatched operands, setting them to `null`. For Load, Store, Phi, GEP, and Cmp, it simply copies values from the vector pack variants' own data structure to `OrderedValues`, creating a starting point for later processing. Reduction has only one value, and Gamma places only Phi nodes contained in it to the `OrderedValues`.

3.3 computeCost

This step is self-contained. The cost is either read from an intrinsic guide or estimated using LLVM `TargetTransformInfo` (primarily for load and store).

4 Vector Pack Context

A vector pack context is a data structure that maintains a bidirectional map between values and integers, enabling the use of a bitmap to record a set of values. It is an intra-function analysis.

5 Vector Pack Set

A vector pack set is an abstraction that manages a set of compatible vector packs and is responsible for lowering a set of packs to LLVM IR.

6 Packer

6.1 Load & Store

Consecutive loads need to be packed into a load pack, and consecutive stores need to be packed into a store pack. VeGen defines AccessLayoutInfo to store the analysis results of consecutive memory accesses. It groups a bunch of consecutive accesses into a group, records their offsets from the lowest address access, and defines the lowest access instruction as the leader of the group.

6.2 Reverse Post Ordering

VeGen maintains the reverse post-order traversal of basic blocks within a function using a mapping from LLVM BasicBlock to an unsigned integer, denoted as BlockOrdering.

6.3 The Big Packer