Speculative Dynamic Vectorization for HW/SW Co-designed Processors

Rakesh Kumar
Alejandro Martínez
Antonio González

Introduction/Motivation

Static Vectorization

- Effective for traditional array-based applications
- Conservative to ensure correctness
- Pointers complicate dependence analysis
- It Limits vectorization opportunities

The Proposal

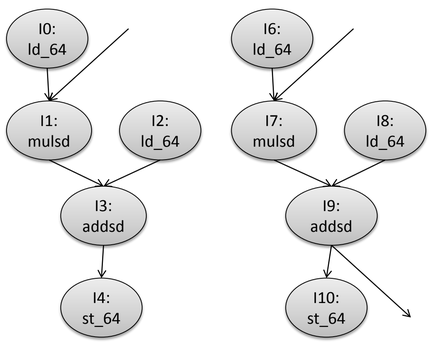
- Vectorize at run time
- Speculatively reorder ambiguous memory accesses
  - It will uncovers more vectorization opportunities
- Hardware checks for any memory violation
  - Interpret the code in case of violation
- Algorithm
  - Consider ambiguous memory references as independent.
    - Convert them to speculative instructions if reordered
  - Vectorize consecutive stores
    - Follow Pred/Succ chains. Generate Pack/Upack if necessary
  - Vectorize remaining consecutive loads
    - Follow Pred/Succ chains. Generate Pack/Upack if necessary
  - Vectorize remaining arithmetic instructions
    - Follow Pred/Succ chains. Generate Pack/Upack if necessary

An Example

The following loop can not be vectorized if the two arrays can not be proved non overlapping statically.

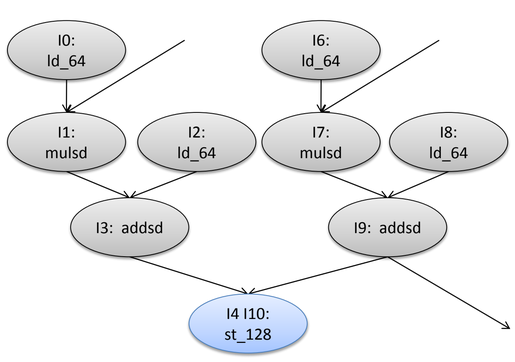
```
void example(double *a, double *b)
{
   int i;
   for (i = 0; i < NUM_ITR; i++)
      a[i] += b[i] * CONST;
}
```

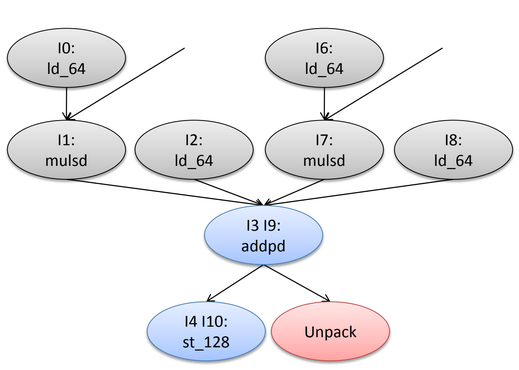
Vectorization using the proposed algorithm:



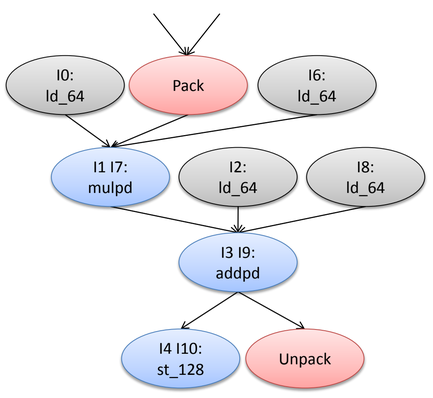
loop:	I0	ld_64	v2, M [r2 + r1 * 8]
	I1	mulsd	v3, v2, v1
	I2	ld_64	v4, M [r3 + r1 * 8]
	I3	addsd	v5, v4, v3
	I4	st_64	v5, M [r3 + r1 * 8]
	I5	add	r4, r1, 1
	I6	ld_64	v6, M [r2 + r4 * 8]
	I7	mulsd	v7, v6, v1
	I8	ld_64	v8, M [r3 + r4 * 8]
	I9	addsd	xmm0, v8, v7
	I10	st_64	xmm0, M [r3 + r4 * 8]
	I11	add	r1, r4, 1
	I12	cmp	r1, r0
	I13	jne	loop

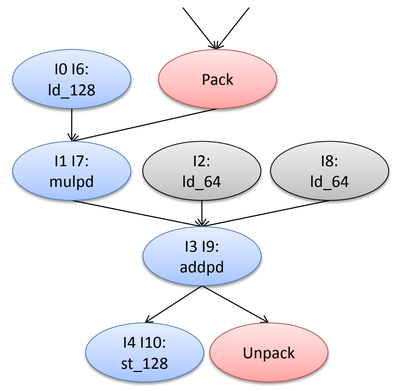
DDG and the low level code for the loop.



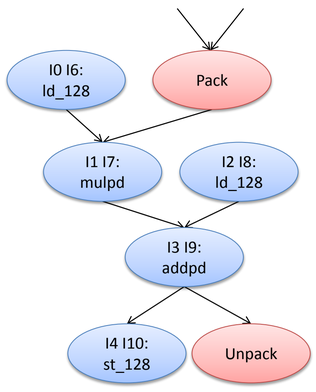


Vectorization continues





Vectorization continues



Loop:	Pack		v12, xmm1, xmm1
	ld_pd_spec		v10, [ebx+eax*8]
	mulpd		v11, v12
	ld_pd _spec		v13, [edx+eax*8]
	addpd		v14, v11, v13
	st_pd _spec		[edx+eax*8], v14
	add		eax,0x2
	cmp		eax,ecx
	jne		loop
	Unpack		xmm0, v14

Vectorization completes: DDG and vectorized code

Speculation and Recovery

Software

- Label instructions in the program order
- If a pair of memory references:
  - may alias and
  - is reordered

convert original load/store instructions to speculative instructions

```
1     ld_64     v1, M[x]
2     st_64     v2, M[y]

2     st_64_spec     v2, M[y]
1     ld_64_spec     v1, M[x]
```

Hardware

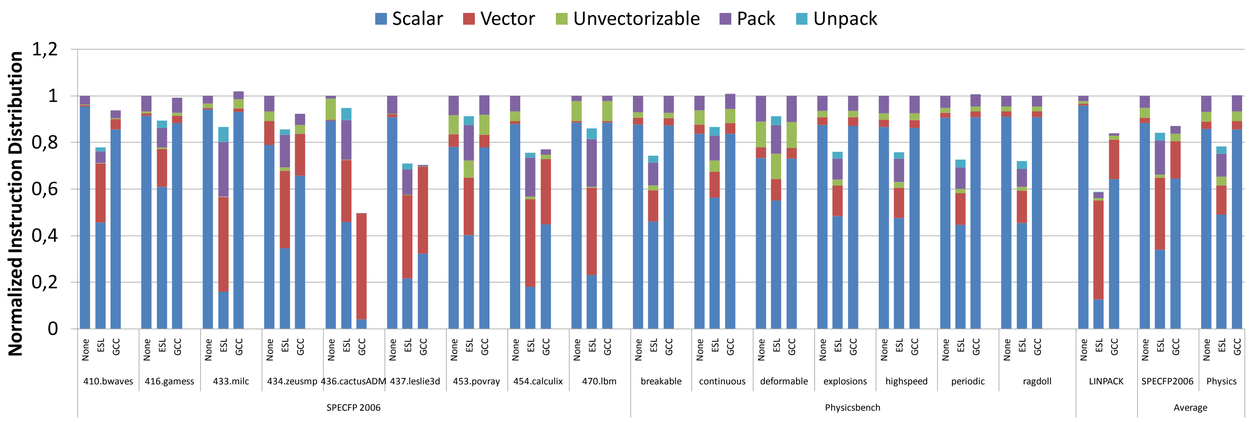
- Save Register and Memory State before executing speculative code

```
PC:  2   st_64_spec   v2, M[y]
PC:  1   ld_64_spec   v1, M[x]
```

Label	Address	Size
2	y	8

- If violation, restore saved state and restart execution without speculation

Results



The proposed algorithm outperforms GCC by 25%, 22% and 3% for LINPACK, Physicsbench and SPECFP2006 respectively.

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

PACT'12
September 19–23, 2012, Minneapolis, Minnesota, USA