KCGRA- Uma Arquitetura Reconfigurável de Dominio Específico para K-means

Matheus Alves, Lucas Bragança, Jeronimo P., **Ricardo Ferreira**, José Nacif - Universidade Federal de Viçosa

CNPq
FAPEMIG
CAPES

Summary

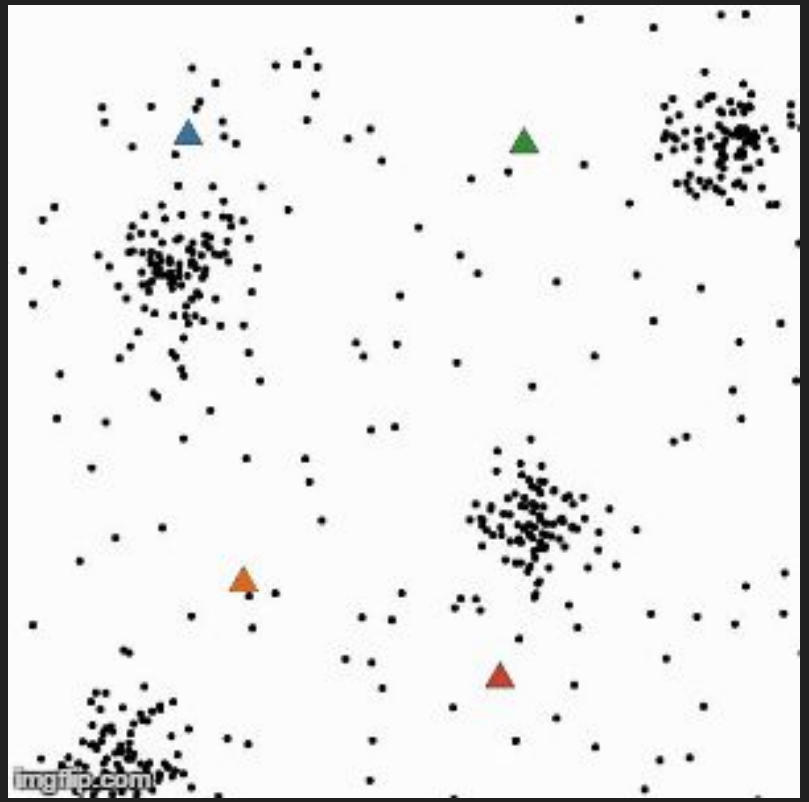
- Kmeans and Hardware Accelerator
- Generic and Specialized Hardware Accelerators
- HP-CGRA and a Specialized Reconfigurable on-the-fly Kmeans Accelerator = **KCGRA**
- Final Considerations

Kmeans and Accelerator

- Well Known Unsupervised Learning Cluster Algorithm
  - NP-Complete
  - Heuristic Complexity O(KN) where K=Clusters and N=attributes

- Most Hardware Accelerators use large values of K (not realistic)
- Which is the best value of K ?
  - FPGA accelerators, K is defined in Design time
  - Hours to compile….

Kmeans Examples

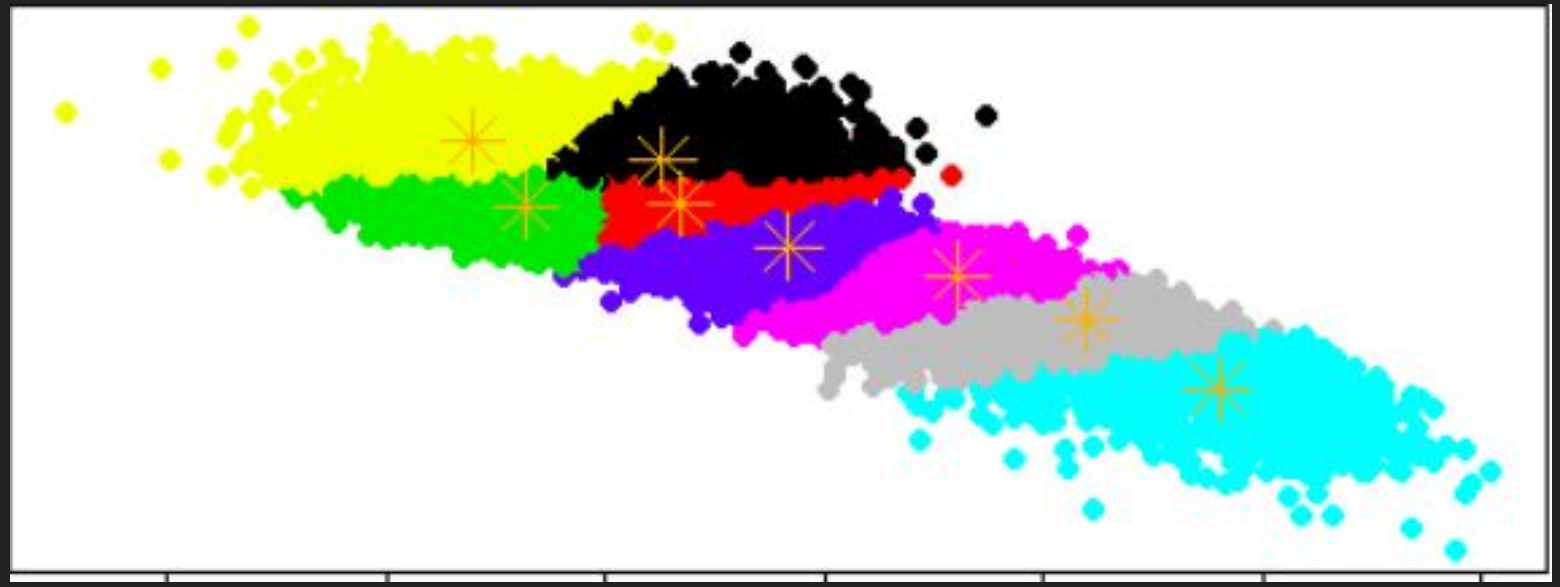


K= 4 clusters

N = 2 Attributes

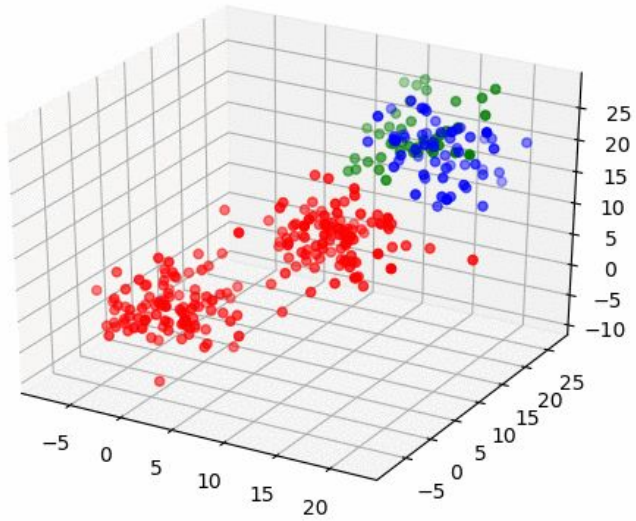
Figure Source

Kmeans Examples



K= 8 clusters N = 2 Attributes

Kmeans Examples



K= 3 clusters

N = 3 Attributes

O(3K) Arithmetic Intensity

Turing Lecture 2018 Patterson&Hennessy

Part III: Future Architecture Opportunities - *Domain Specific Languages and Architecture,*

- Achieve higher efficiency by tailoring the architecture to characteristics of the domain
  - Not one application, but a domain of applications
    - Different from strict ASIC
  - Requires more domain-specific knowledge then general purpose processors need

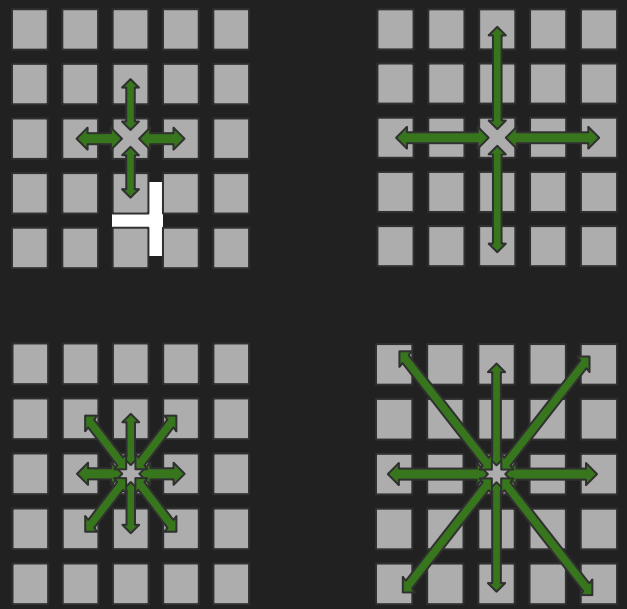
What Opportunities Left?

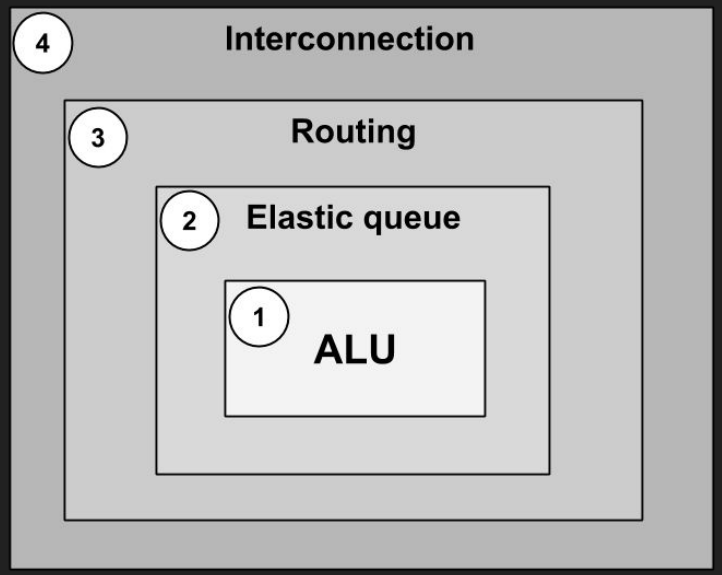
- SW-centric
  - Modern scripting languages are interpreted, dynamically-typed and encourage reuse
  - Efficient for programmers but not for execution
- HW-centric
  - Only path left is *Domain Specific Architectures*
  - Just do a few tasks, but extremely well
- Combination
  - Domain Specific Languages & Architectures

- FPGA
  - Flexibility and bit level ✓
  - Hardware Knowledges ⊖
  - Hours to Compile and Hard to Deploy ✗
- CGRA
  - Simplify the Placement and Routing ⊖
  - Performance ⊖
  - No Commercial CGRA ✗

- HP-CGRA
  - Generator for Commercial FPGA
  - Parametric
  - FPGA Portability - Intermediate Format
  - Simplify Placement & Routing
- **Proposal Specialized K-CGRA**
  - Extension HP-CGRA
  - New More Coarse-Grained Operators
  - Performance close to FPGA implementation
  - K value is at runtime

HP-CGRA Parametric Generator

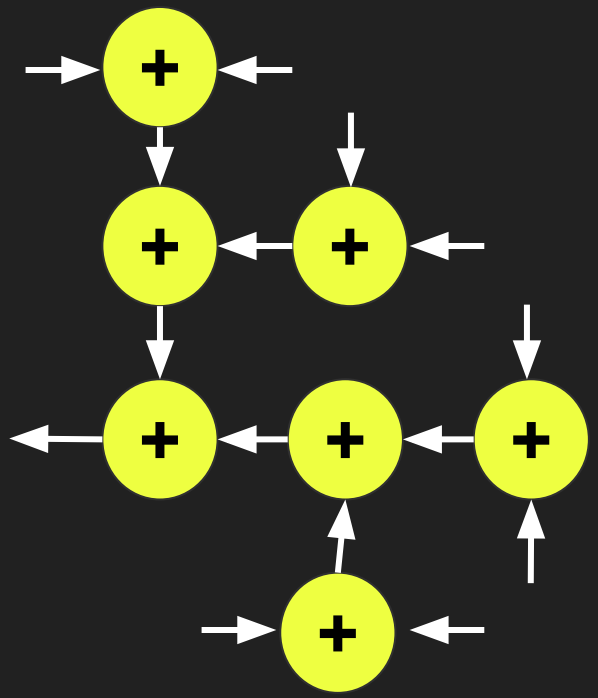




HP-CGRA Parametric Generator

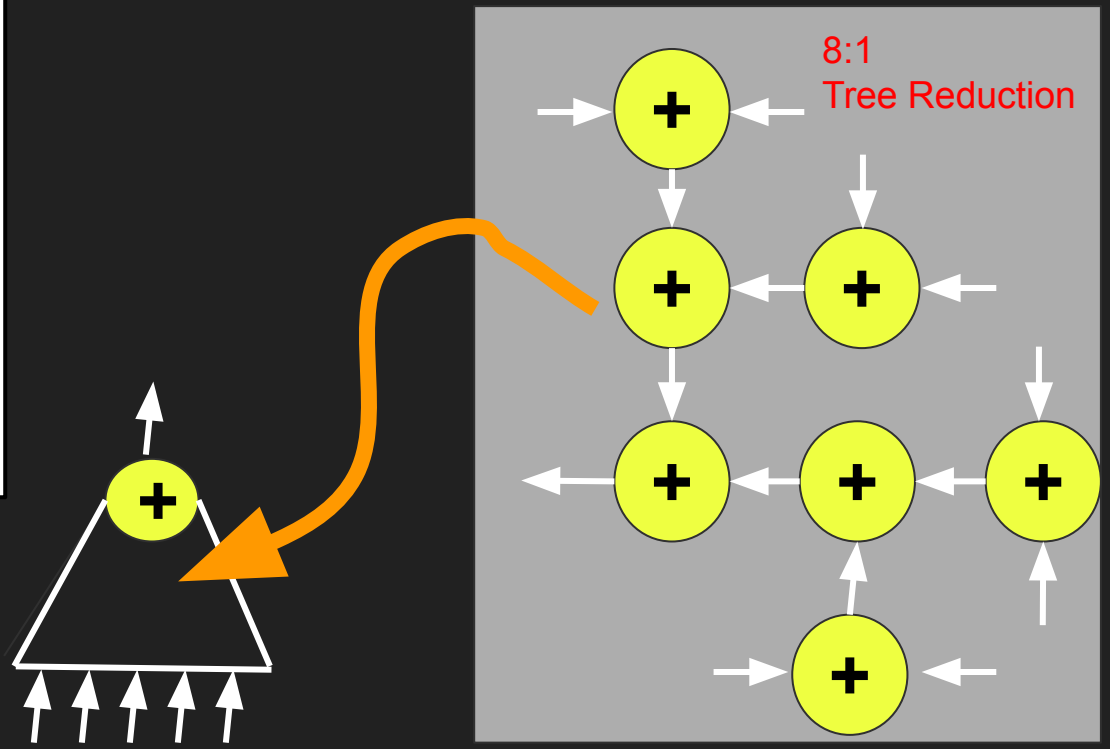
```
{
    "shape": [ 3,  3],
    "data_width": 8,
    “pes”: [
        :
        :
    ]
}
```

```
{
    “id”: 0,
    “type”: “basic”,
    “neighbors”:[1, 3],
    “route_type”: “one_routing”,
    “elastic_queue”: 2,
    “isa”: [“and”, “mux”, “madd”]
}
```

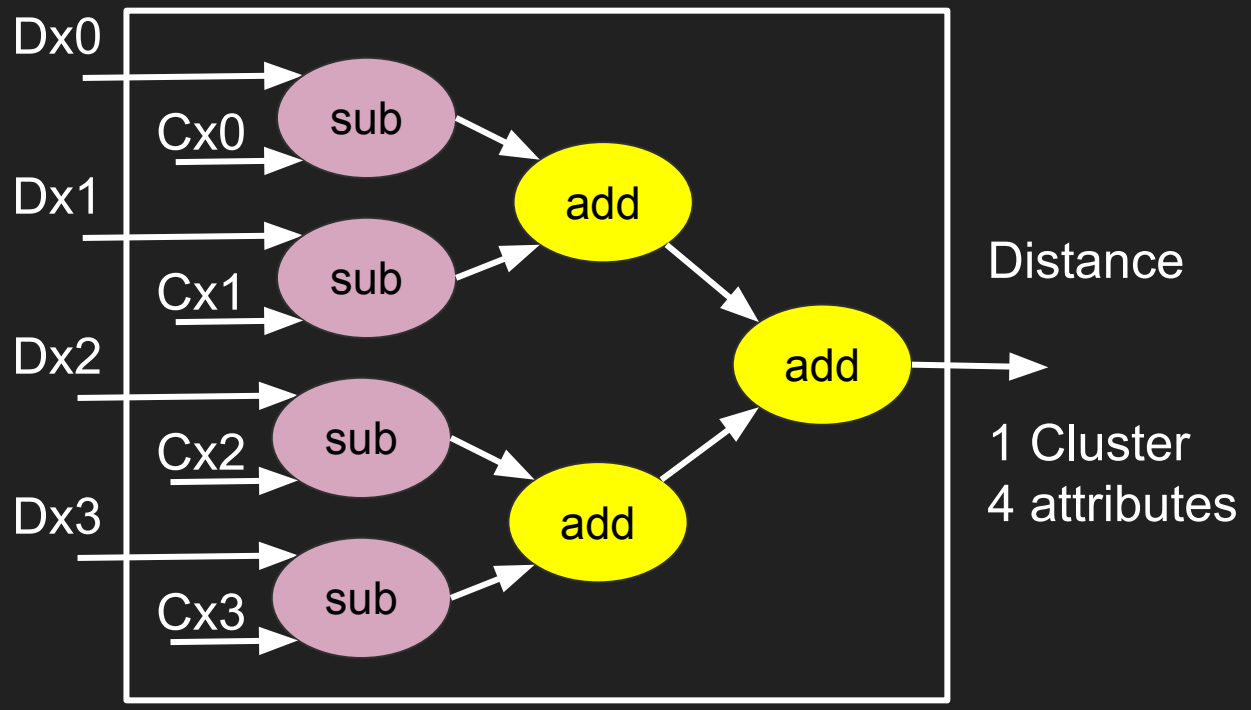


Extension: composite coarse-grained operators

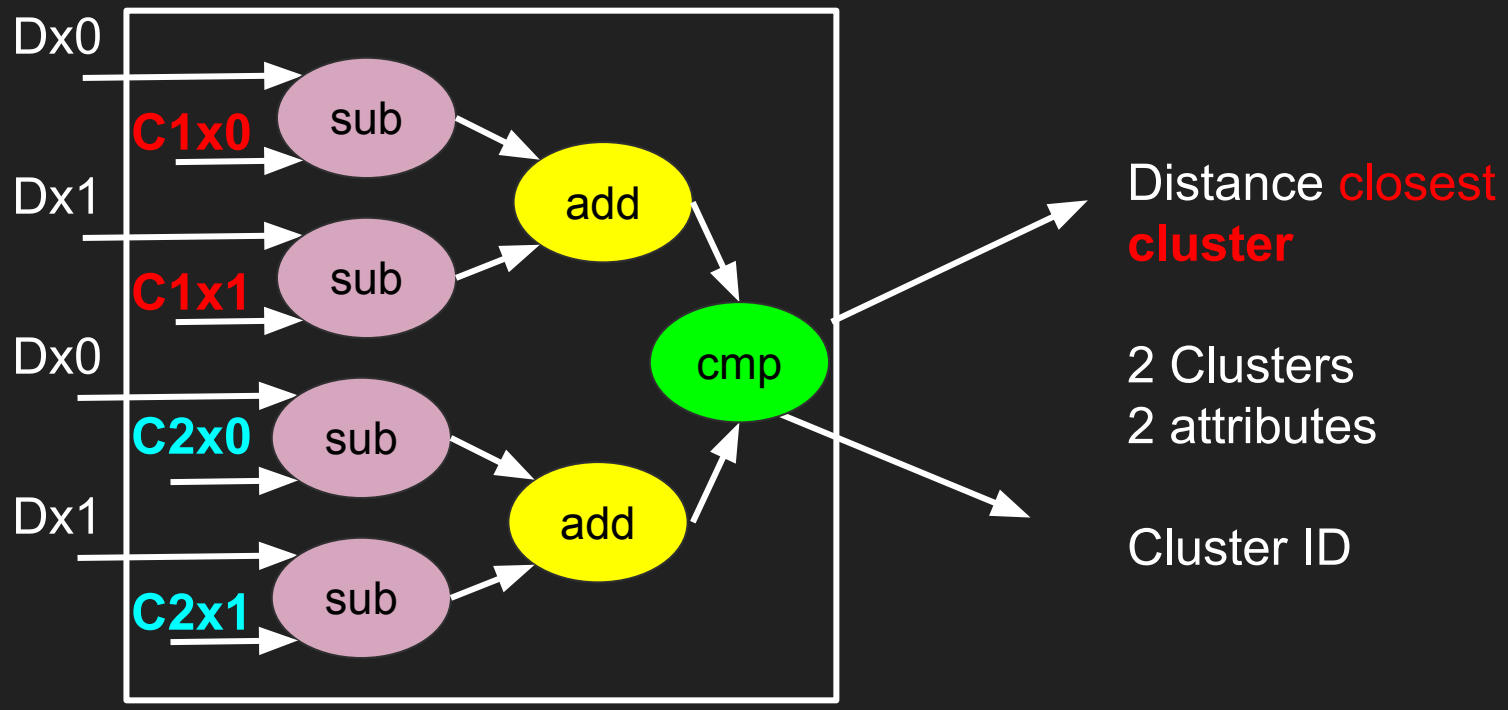
```
{
    "id": 0,
    "type": "basic",
    "neighbors":[1, 3],
    "route_type": "one_routing",
    "elastic_queue": 2,
    "isa": [ reduction_operator]}
```



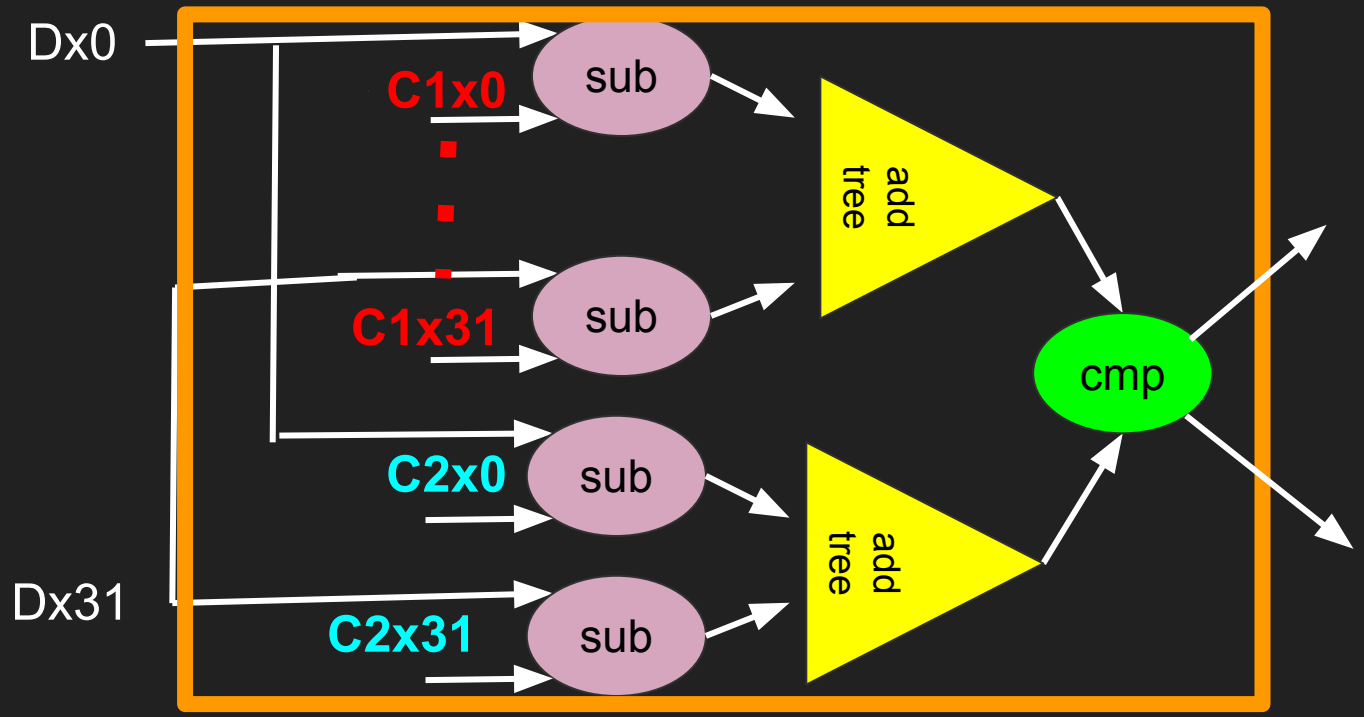
Reconfigurable Distance Operator



Reconfigurable Distance Operator



Reconfigurable Distance Operator 2x32

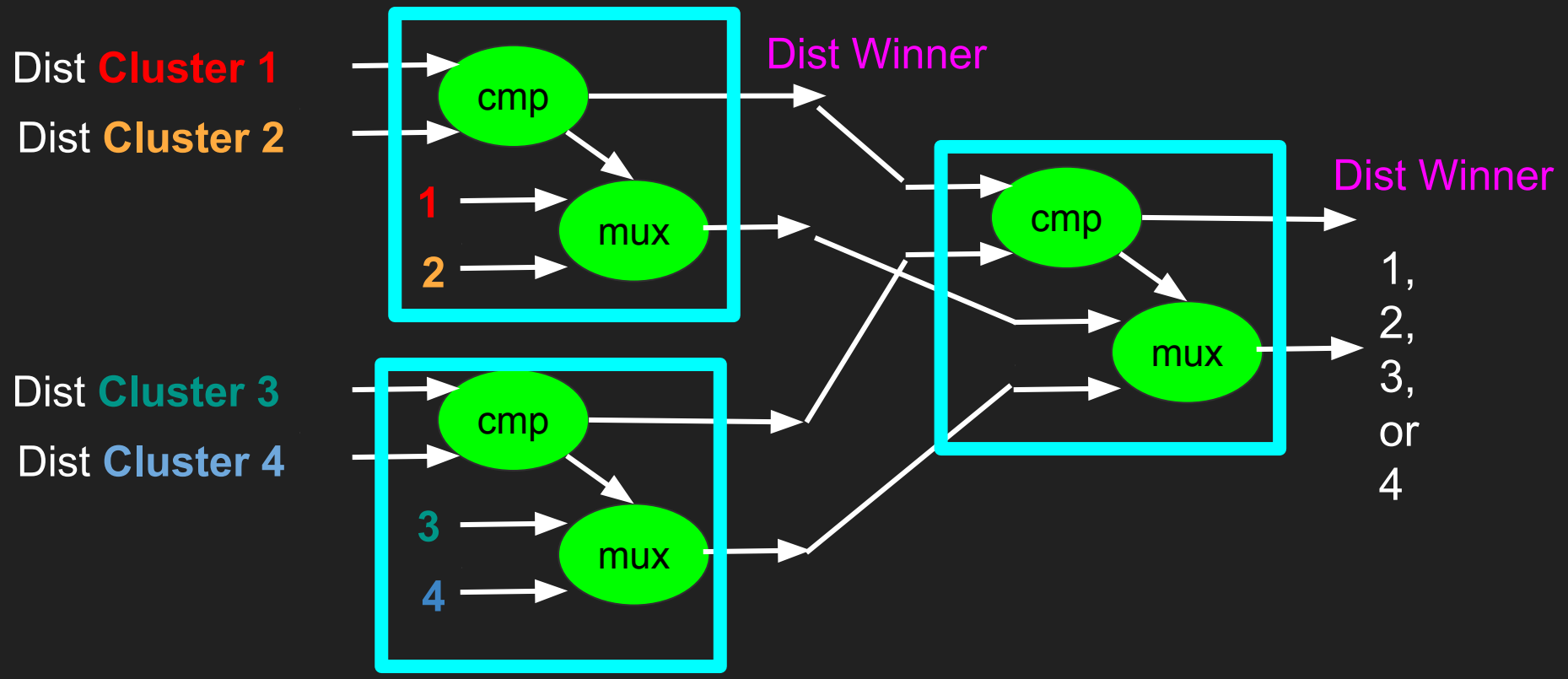


Distance closest cluster

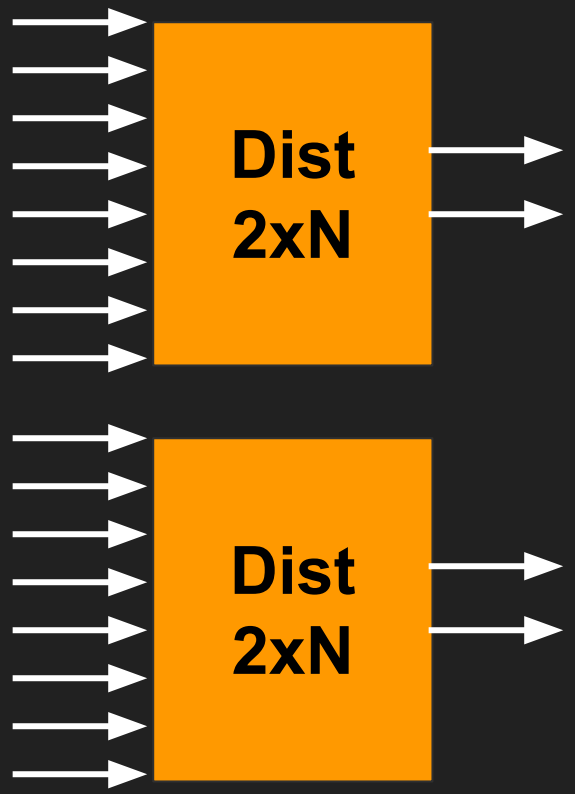
2 Clusters
32 attributes

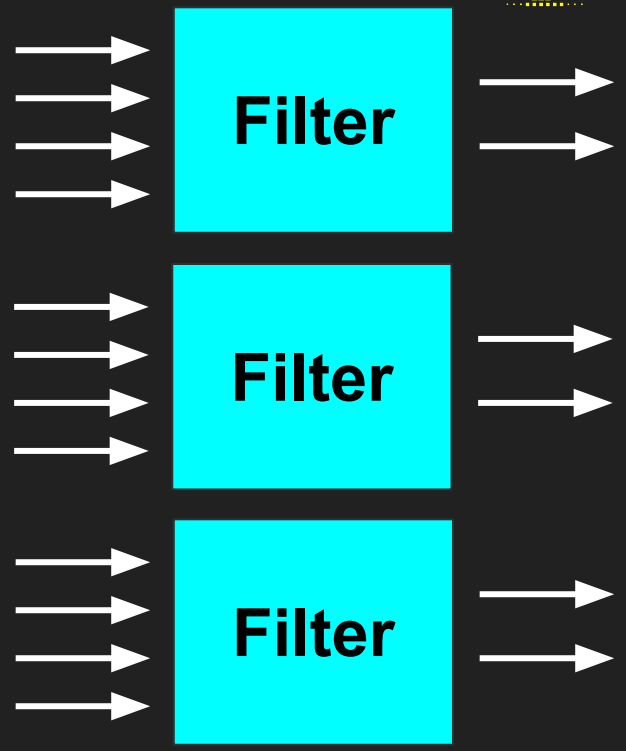
Cluster ID

Reconfigurable Filter Operator 2x

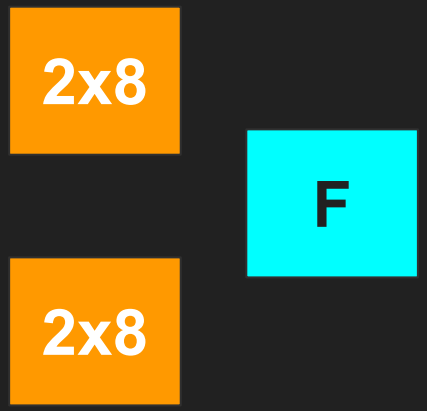


Reconfigurable K CGRA

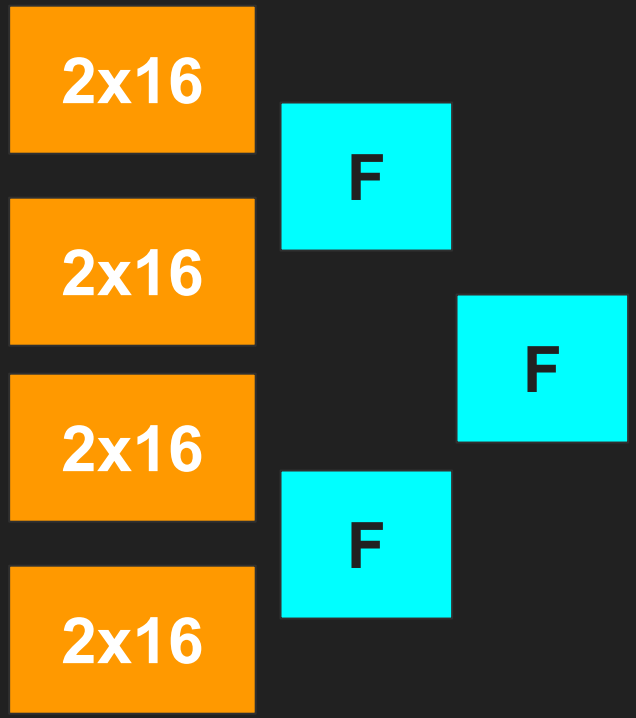




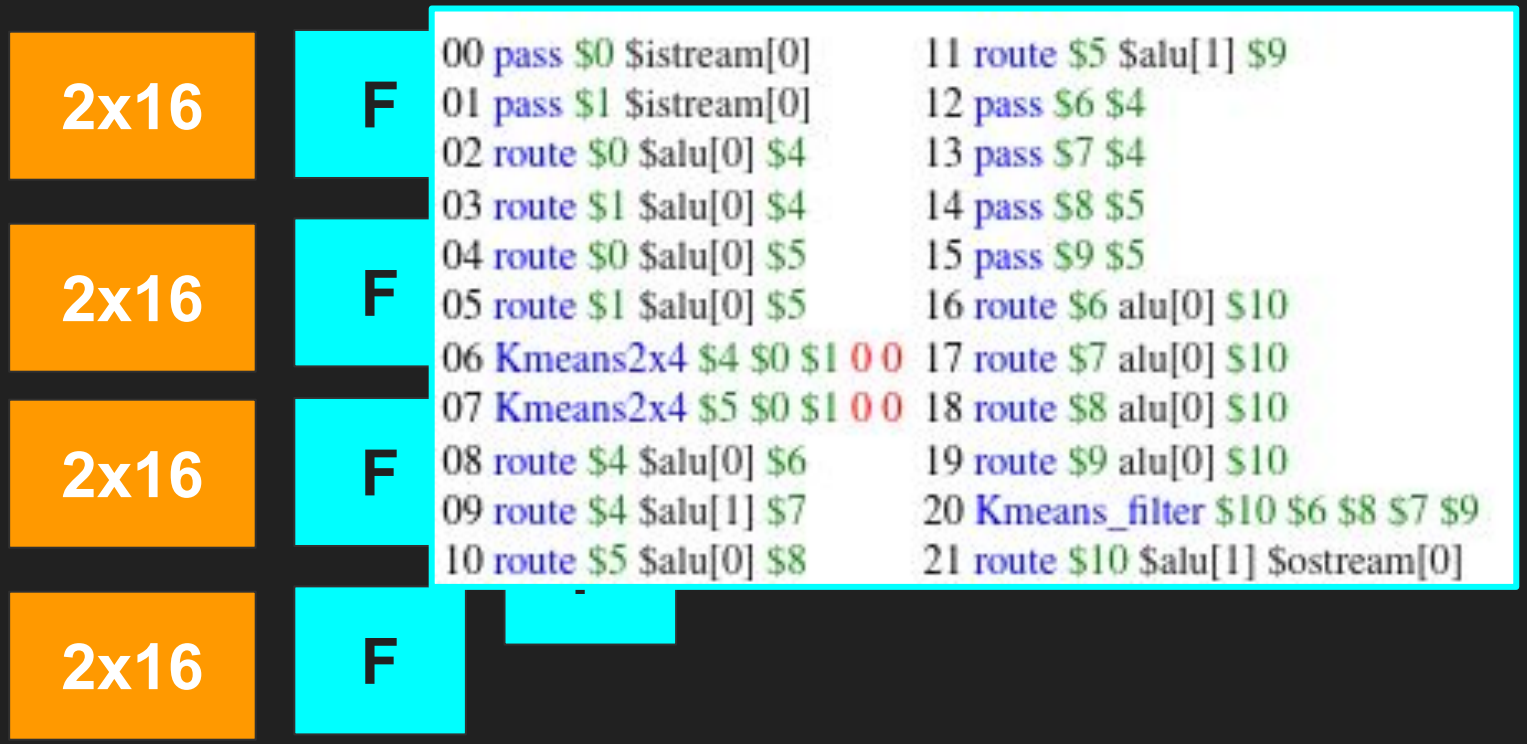
Reconfigurable K CGRA
K=4 N =8



Reconfigurable K CGRA
K=8 N =16



Assembly Reconfigurable K CGRA

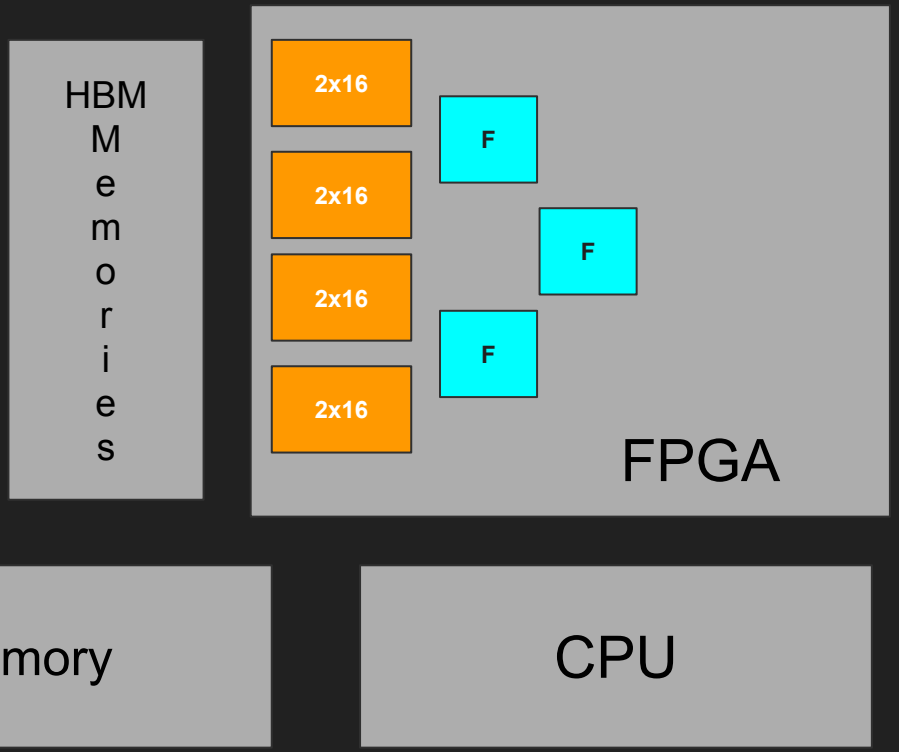



```
00 pass $0 $istream[0]
01 pass $1 $istream[0]
02 route $0 $alu[0] $4
03 route $1 $alu[0] $4
04 route $0 $alu[0] $5
05 route $1 $alu[0] $5
06 Kmeans2x4 $4 $0 $1 0 0
07 Kmeans2x4 $5 $0 $1 0 0
08 route $4 $alu[0] $6
09 route $4 $alu[1] $7
10 route $5 $alu[0] $8
11 route $5 $alu[1] $9
12 pass $6 $4
13 pass $7 $4
14 pass $8 $5
15 pass $9 $5
16 route $6 alu[0] $10
17 route $7 alu[0] $10
18 route $8 alu[0] $10
19 route $9 alu[0] $10
20 Kmeans_filter $10 $6 $8 $7 $9
21 route $10 $alu[1] $ostream[0]
```

Assembly Reconfigurable K CGRA on High Performance Alveo FPGAs

WSCAD 2023

```
00 pass $0 $istream[0]
01 pass $1 $istream[0]
02 route $0 $alu[0] $4
03 route $1 $alu[0] $4
04 route $0 $alu[0] $5
05 route $1 $alu[0] $5
06 Kmeans2x4 $4 $0 $1 0 0
07 Kmeans2x4 $5 $0 $1 0 0
08 route $4 $alu[0] $6
09 route $4 $alu[1] $7
10 route $5 $alu[0] $8
11 route $5 $alu[1] $9
12 pass $6 $4
13 pass $7 $4
14 pass $8 $5
15 pass $9 $5
16 route $6 alu[0] $10
17 route $7 alu[0] $10
18 route $8 alu[0] $10
19 route $9 alu[0] $10
20 Kmeans_filter $10 $6 $8 $7 $9
21 route $10 $alu[1] $ostream[0]
```



HLS versus RTL versus KCGRA

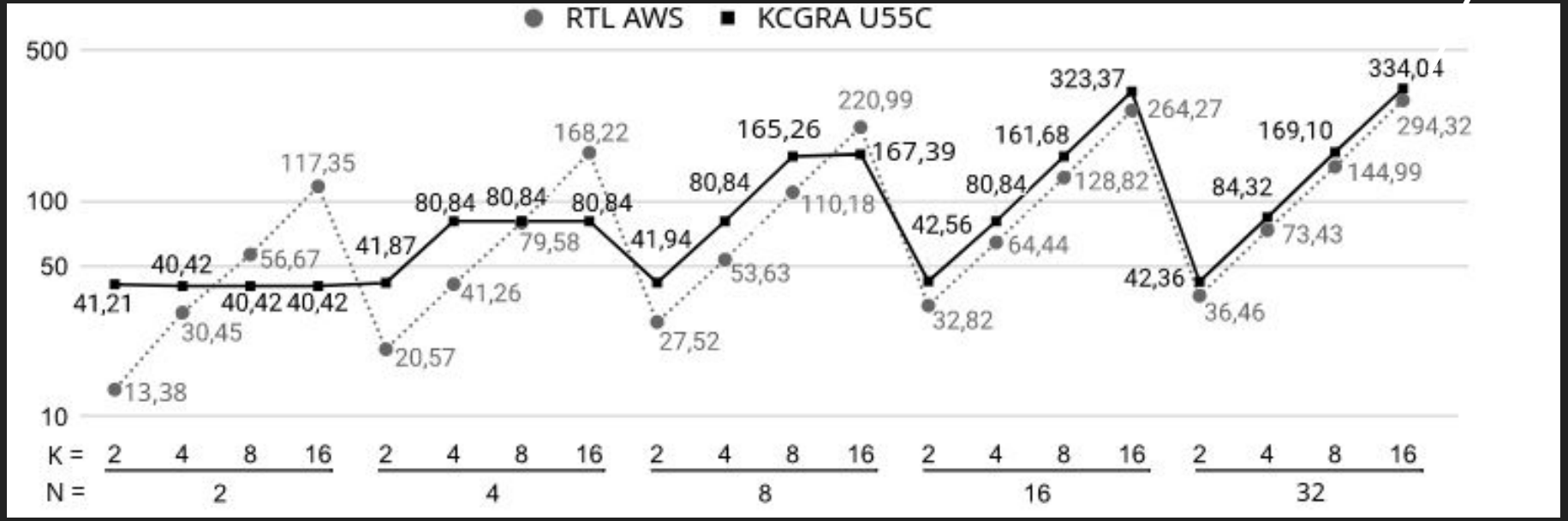
WSCAD 2023

	K	N	CPU->FPGA	FPGA->CPU	Tkernel	Accel
HLS	4	8	5,89	0,84	7,20	1
	8	8	6,26	0,92	7,20	1
	16	8	7,27	1,05	7,36	1
RTL	4	8	2,93	0,37	3,27	2,20
	8	8	2,84	0,33	3,44	2,09
	16	8	2,85	0,37	3,30	2,23
KCGRA	4	8	4,04	0,55	2,32	3,11
	8	8	3,54	0,54	2,32	3,10
	16	8	7,16	1,11	4.59	1,60

RodiniaHLS [Cong and et al. 2018]
RTL [Bragança et al. 2021

RTL AWS versus KCGRA U55C

668 Gops



HLS versus RTL versus KCGRA FGPA Resources

	K	N	LUT	Lut as Mem	Reg	Bram	DSP	Reconfiguration
HLS	4	8	8.241	1.581	14.152	119	96	03h 05m 45s
	8	8	10.501	1.467	16.637	119	192	03h 10m 21s
	16	8	16.579	2.253	22.617	119	384	03h 11m 10s
RTL	4	8	9.350	306	11381	7	384	02h 44m 25s
	8	8	17.632	318	21.768	7	768	02h 56m 28s
	16	8	34.361	402	40.041	7	1.536	03h 23m 53s
KCGRA	2..32	2..32	153.494	6298	158497	0	0	2,34ms

KCGRA versus related Work

	Reconfig	Implem	Validation	Peak Gops	Data Size
[Lopes et al. 2017]	Yes	-	Simulation		10k
[Tang and Khalid 2016]	No	OpenCL	Simulation	150	2M
[Paulino et al. 2020]	No	OpenCL	Simulation	50	4k
[Penha et al. 2018]	No	RTL Gen	Execution	40	2M
[Dias et al. 2020]	No	RTL	Simulation	4	4k
[Bragança et al. 2021]	No	RTL Gen	Execution	220	2M
[Gorgin et al. 2022]	No	RTL	Simulation		
	Yes	JSON CGRA	Execution	668	2M

Turing Lecture 2018 Patterson&Hennessy

Conclusion: A New Golden Age

- Domain Specific Languages ⇒ Domain Specific Architectures
- Free, open architectures and open source implementations ⇒ everyone can innovate and contribute
- Cloud FPGAs ⇒ all can design and deploy custom “HW”

- DSL by using SW-HW generators
- DSA by extending HP-CGRA, example Kmeans
- Portability by using FPGAs overlay and intermediate format
- Performance and Deploy in Cloud FPGA

Questions ?

ricardo@ufv.br

Acknowledgments

Financial support from FAPEMIG APQ-01577-22, CNPq, and UFV. This work was also carried out with the support of the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Financing Code 001