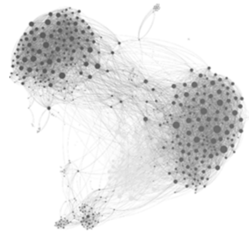
MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

MPSoC'2015
July 13-17, 2015
Ventura, CA

An Embedded Computing Architecture for Finding Similarities in Large Networks

HPC on embedded computing devices
- a Case Study -



TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Norbert Wehn
wehn@eit.uni-kl.de

MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

General Trends

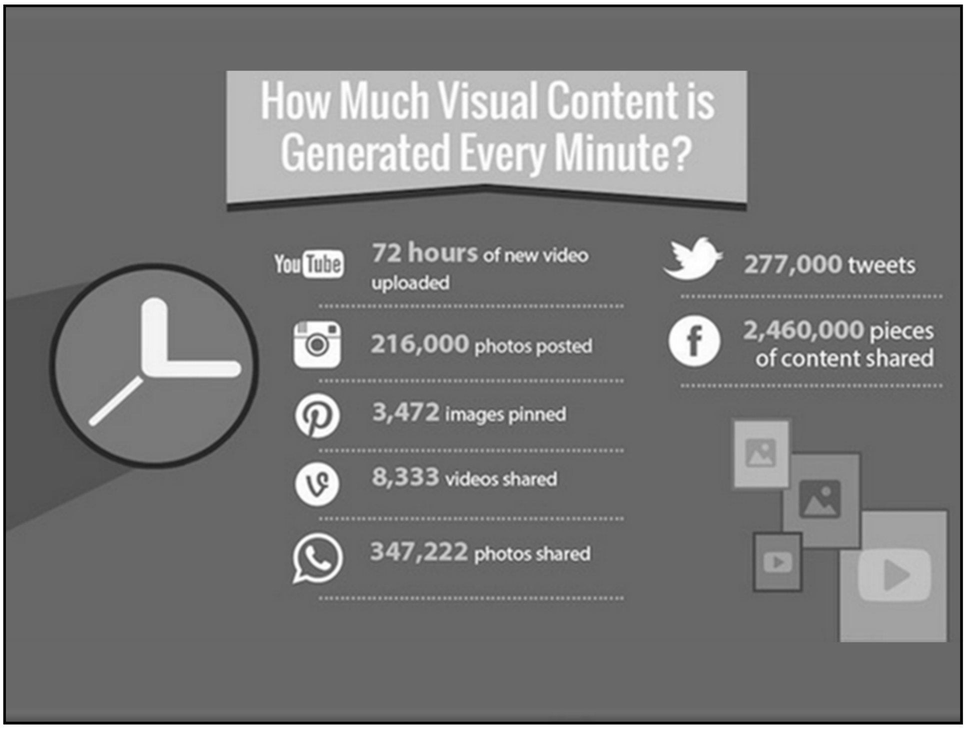
Many Big Data applications

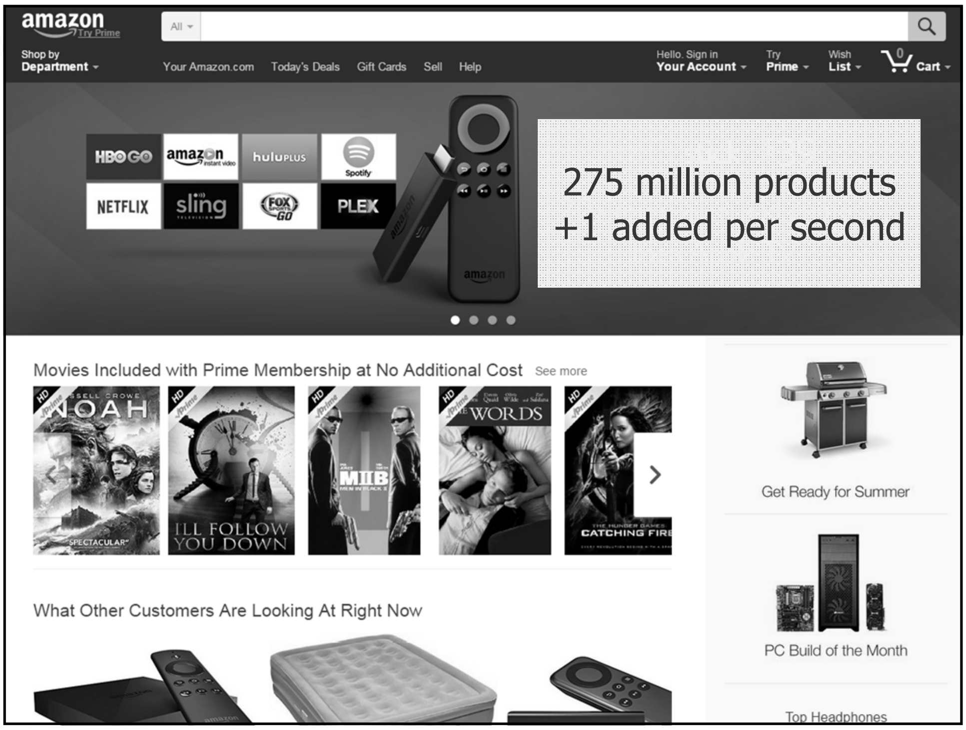
- Fast and reliable identification of so-called network motifs in large networks
- Subgraphs whose occurrence is significantly higher than expected in a random graph model
- Here special variant of motifs: co-occurrence
- Application examples: cleaning biological data, e-commerce like recommendation systems

HW architectural trends

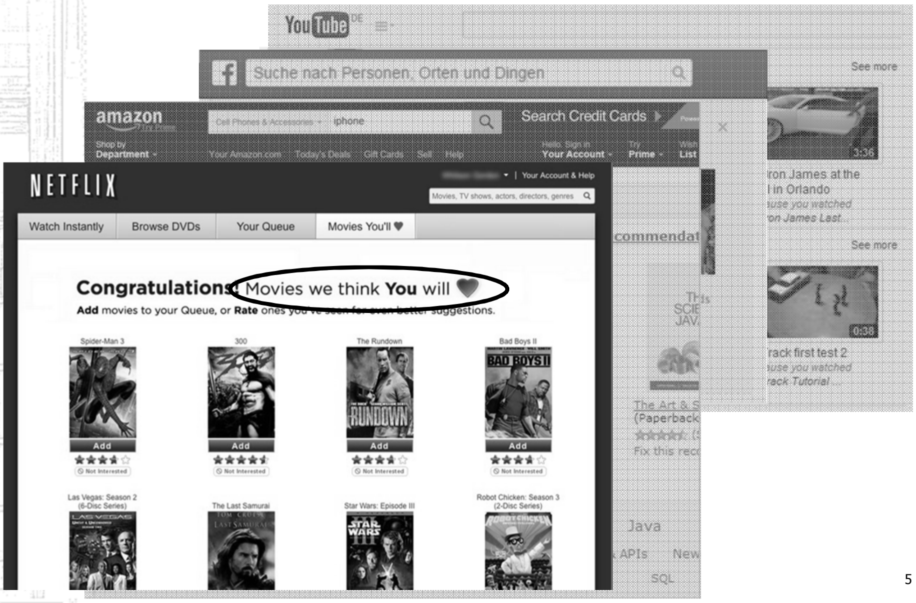
- More and more functionality moves into embedded devices
- Dedicated accelerator approach due to energy efficiency

2


How Much Visual Content is Generated Every Minute?
YouTube
72 hours of new video uploaded
216,000 photos posted
3,472 images pinned
8,333 videos shared
347,222 photos shared
277,000 tweets
2,460,000 pieces of content shared

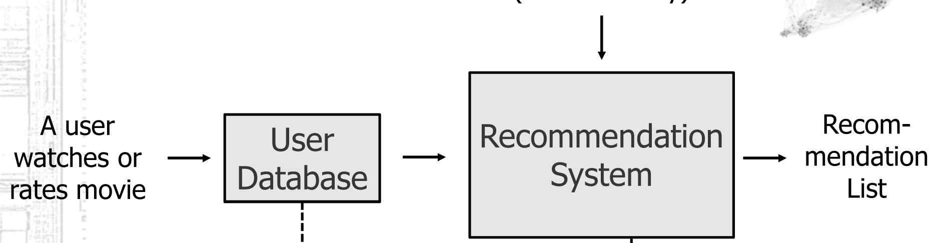

amazon
Try Prime
All
Shop by Department
Your Amazon.com
Today's Deals
Gift Cards
Sell
Help
Hello. Sign in Your Account
Try Prime
Wish List
0
Cart
HBO GO
amazon instant video
huluPLUS
Spotify
NETFLIX
sling
FOX SPORTS GO
PLEX
275 million products
+1 added per second
Movies Included with Prime Membership at No Additional Cost
See more
NOAH
I'LL FOLLOW YOU DOWN
MIIB
THE WORDS
Get Ready for Summer
PC Build of the Month
What Other Customers Are Looking At Right Now
Top Headphones

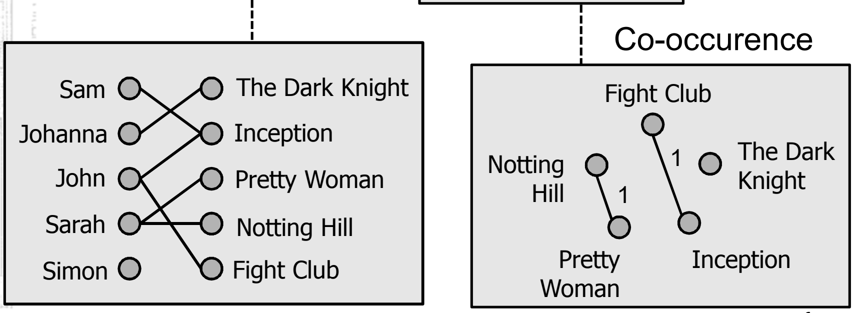
Personalized Recommendation



5

How it works





6

MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

Algorithm

- Absolute co-occurrence value has no real meaning since popular movies will have a high co-occurrence
- Some normalization necessary

Our approach: Markov Chain Monte Carlo based algorithm

1. Calculate expected co-occurrence on a set of random graphs with the same degree distribution (random graph are generated via swaps)

1. Calculate leverage as difference between original co-occurrence and expected co-occurrence: high leverage ⇒ the more similar are the nodes

7

MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

MCMC Algorithm

Netflix: 500k nodes, 50 million edges, 200 MB

Data: Graph $G((V_l, V_r); E)$ with vertices V_l and V_r and edges E, V_l being the vertices of interest;
Result: Leverage, p-value, z-score for all pairs of vertices $(u, v) \in (V_l \times V_l)$;
Calculate $coocc(u, v)\ \forall\ (u, v) \in (V_l \times V_l)$; $G_0 := G$;
for i := 1 to $|samples|$ **do** ← 10^4 sample-graphs
  $G_i := G_{i-1}$;
  Swap randomization:
  for $|swaps|$ **do** ← 10^9 swaps
    Choose two edges at random in G_i and swap them, if no duplicate edge arises from the swap;
  end
  Coocc computation:
  Calculate $coocc_i(u, v)\ \forall\ (u, v) \in (V_l \times V_l)$; ← 10^{12} cooccs
end
Calculate leverage

10^{16} coocc-op.
1 GHz
115 days

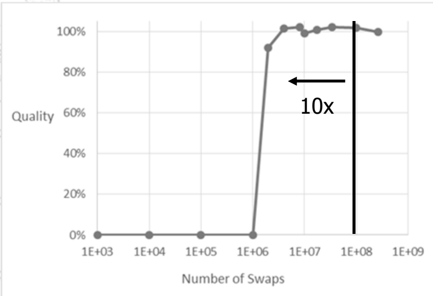
Challenges

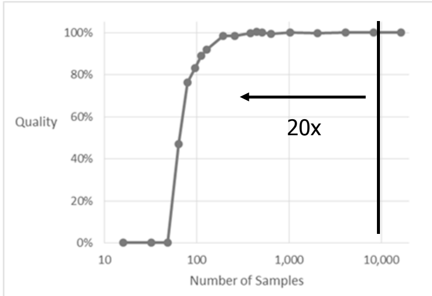
- #samples, #swaps
- Parallelization and efficient coocc calculation
- Coocc computation: graph access, comparison, addition

8

Phase transition

- Huge amount of simulations: observed phase transitions
- Efficient on-line Heuristic: #samples, #swaps





0.9 % Quality Loss

Efficient Coocc-Computation

Graph access is key for fast coocc computation
- Sparse graph e.g. Netflix dataset = 0.6%
- How to store graph: list or full matrix?

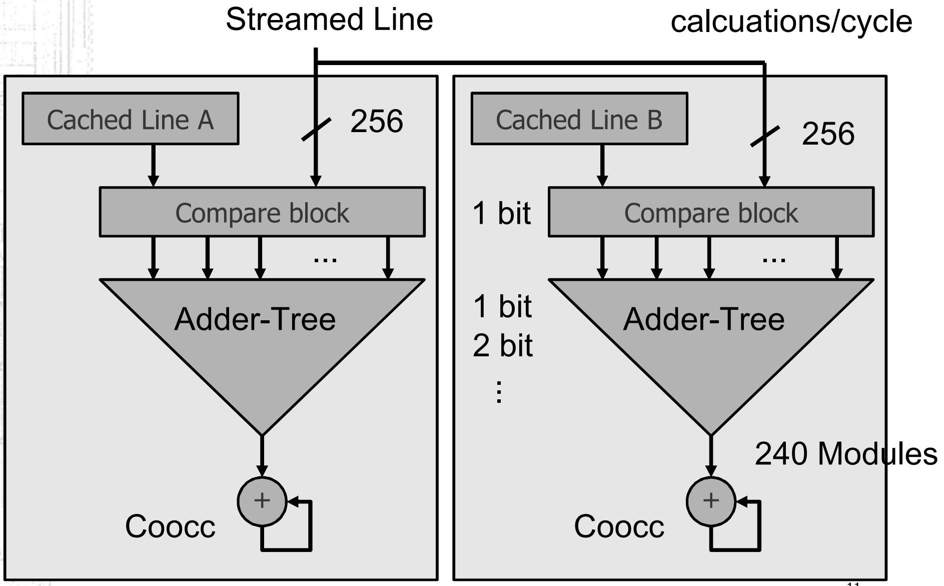
Due to sparse matrix: list is normally the preferred solution (19 bit/entry)

Our approach: store full matrix
- Only 1 bit entry
- Netflix data set ~ 1.2GB
- Very fast access and large parallelism
- 64bit DDR channel@800MHz: 256 edges/cycle

Cache line
Combine comparison and addition
Fully highly adder tree (superior to CPU/GPU)

Coocc Unit

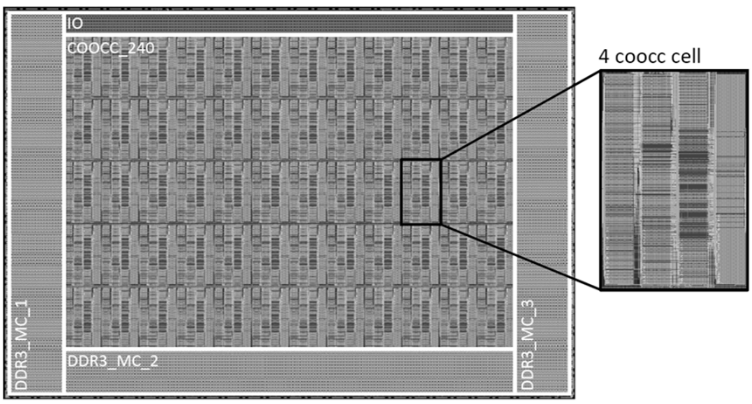
61,440 cooc calcuations/cycle



28 nm ASIC, 400 MHz, 51 mm²

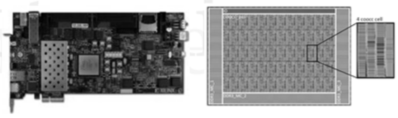
Memory:
3x DDR3,
48 GB,
800 MHz,
38 GB/s

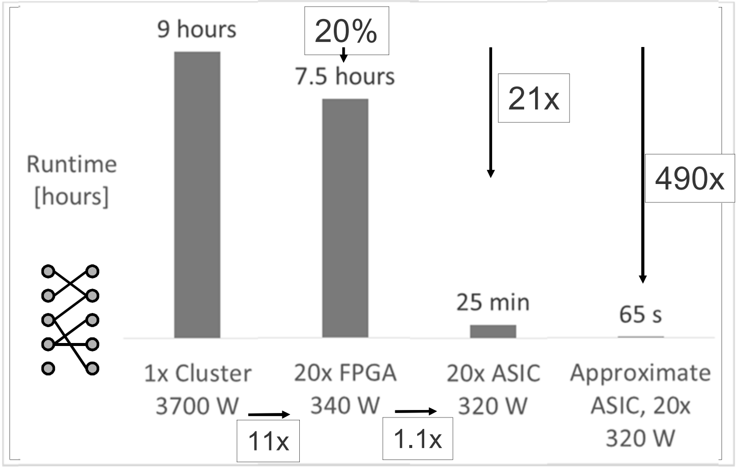
Limitations:
10^7 nodes
10^{11} edges



240 coocc modules, 256 bit adder trees → 61,440 graph coocc-operations/cycle

Results Netflix





Cluster: 10 nodes, each: 2xIntel Xeon X5680 @ 12x 3.33 GHz, 32nm, 48 GB DDR3 memory
FPGA: Xilinx Zynq 7045, 28nm, 8 GB memory, ZC706 development board, 17 W each
ASIC: 28nm low-power bulk, 400 MHz, 48 GB memory, 15.8 W each

13

For more information please visit

http://ems.eit.uni-kl.de