Two More Strategies to Speed Up Connected Components Labeling Algorithms

Federico Bolelli, Michele Cancilla, Costantino Grana
University of Modena and Reggio Emilia

Problem Statement

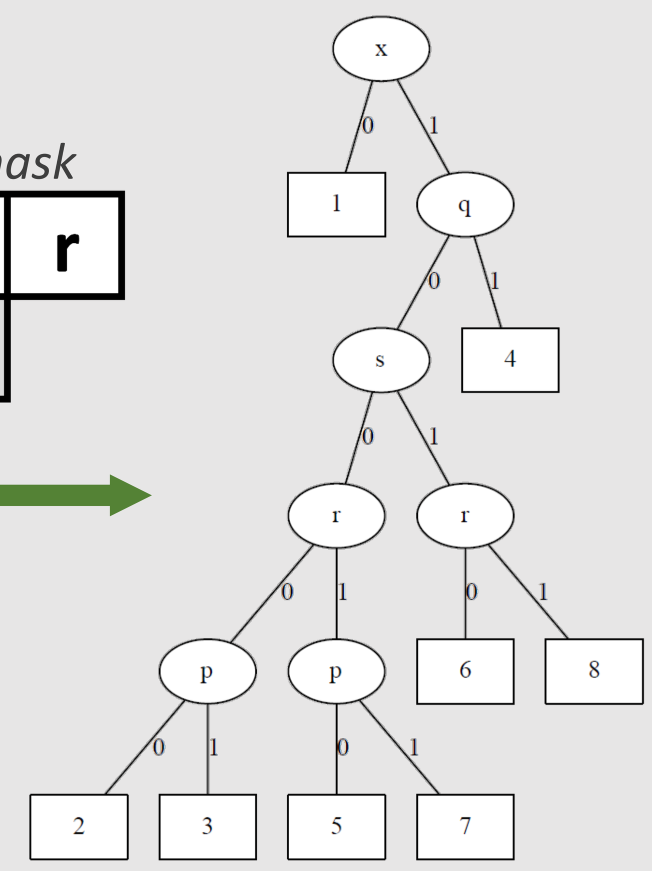
The problem of labeling the connected components (CCL) of a binary image is well-defined and it is a fundamental task in image processing and computer vision applications. Most of recent proposals exploit a scan mask to perform CCL and focus on performance optimization making use of optimal decision trees that allow a reduction of memory accesses.

Goals:

1) **Improve the speed of CCL taking into account image patterns occurrences and altering optimal decision trees;**
2) **Speed up CCL operations using parallelization.**

SAUF mask

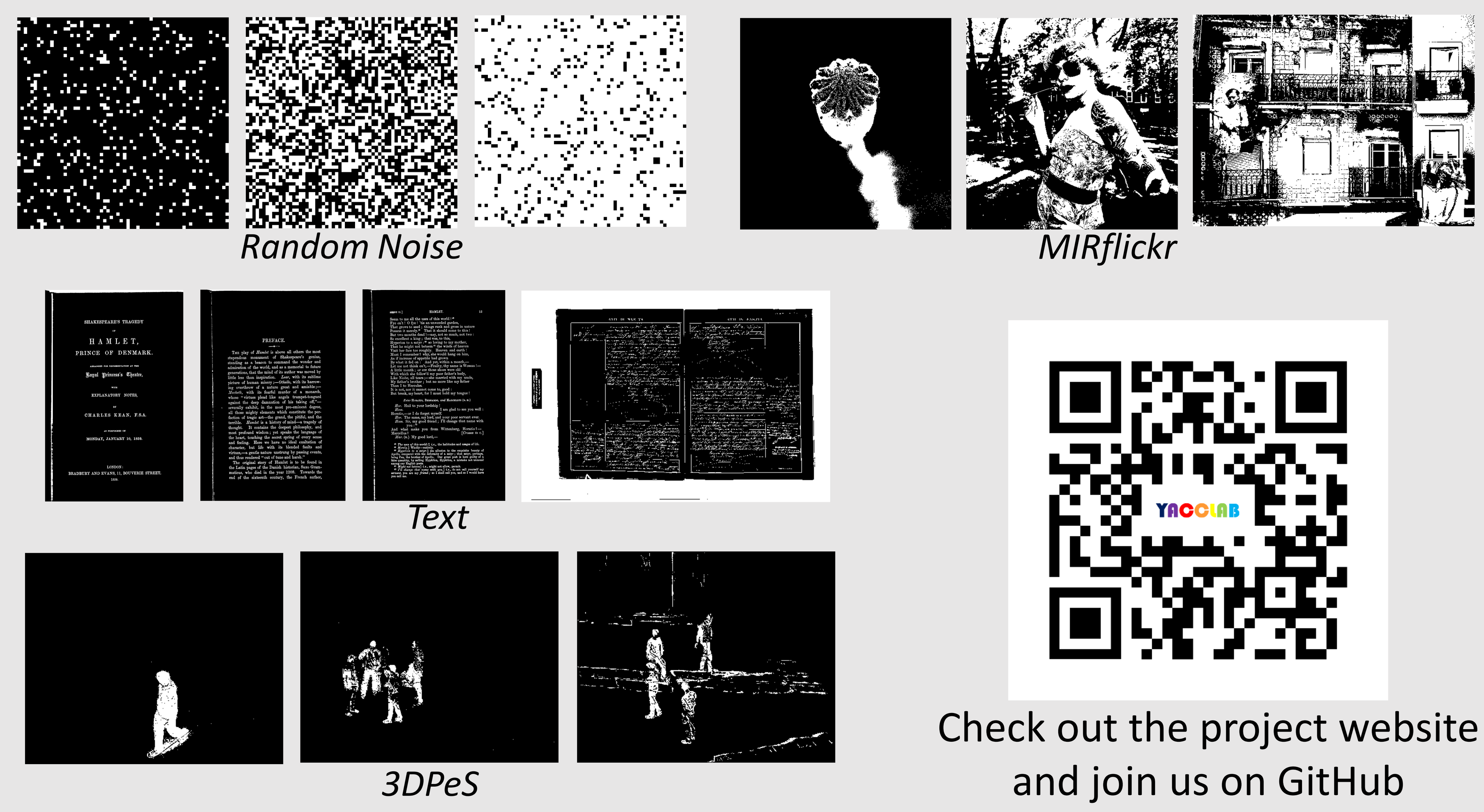
p	q	r
s	x	



The Testing Framework

All reported results are conducted with YACCLAB: an open-source C++ framework for CCL performance evaluation which includes both synthetic and real images.

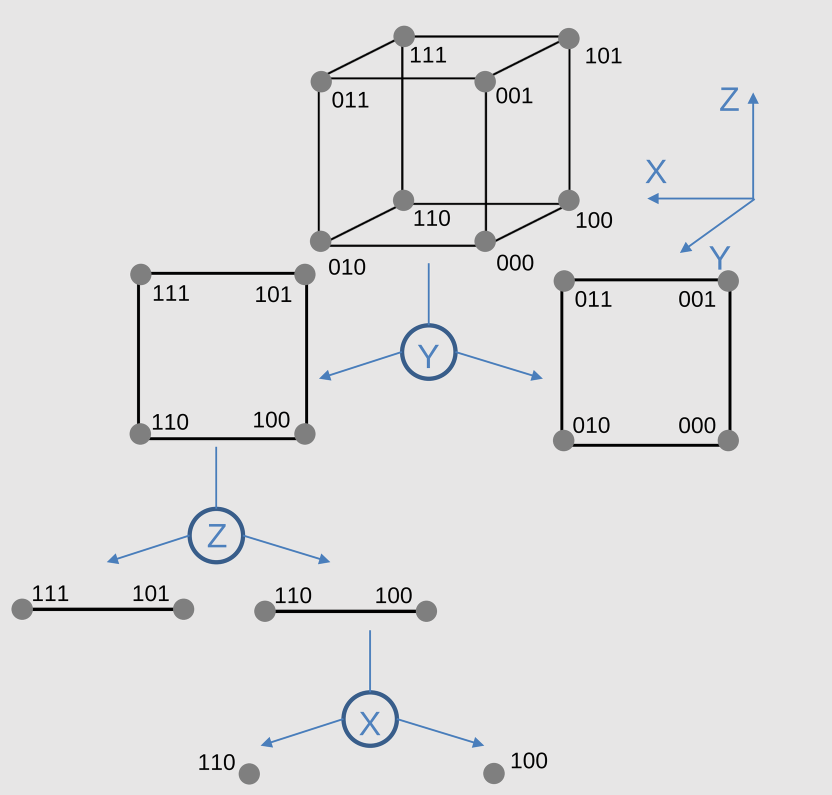
The YACCLAB dataset



Check out the project website and join us on GitHub

Modeling of Decision Trees with Patterns Frequencies

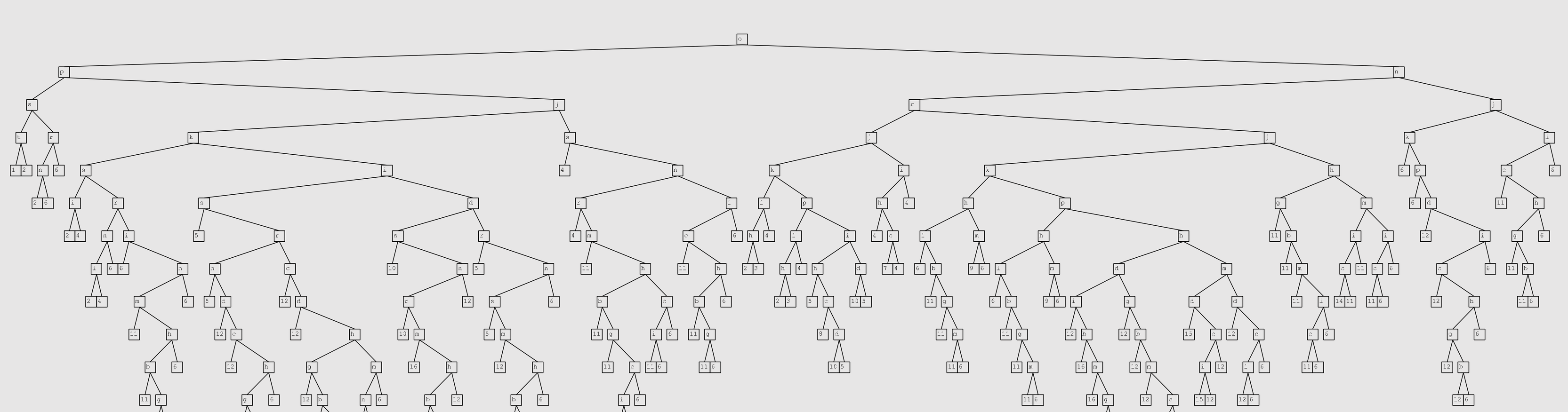
Creation of an optimal decision tree: if two branches lead to the same action, the condition from which they originate may be removed. This conversion can be geometrical interpreted as the partitioning of an n-dimensional hypercube where the vertexes correspond to the 2^n possible rules. Associating to each condition removal a unitary gain we can select the tree which maximizes the total gain and thus minimizes the number of memory accesses.



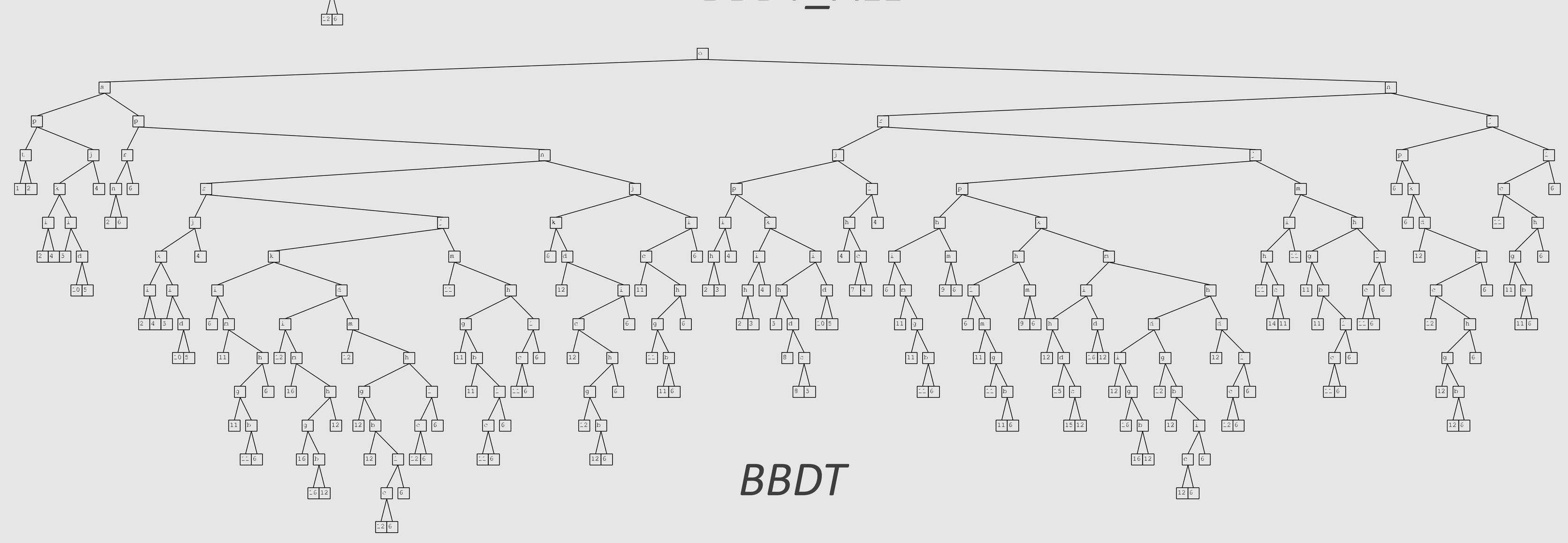
a	b	c	d	e	f
g	h	i	j	k	l
m	n	o	p		
q	r	s	t		

BBDT mask

Idea: We calculate the occurrence frequencies of all possible mask patterns in a reference dataset. Then, considering as gain of a condition removal the frequencies of associated patterns, it is possible to generate new optimal decision trees, further reducing the total number of memory accesses.



BBDT_ALL

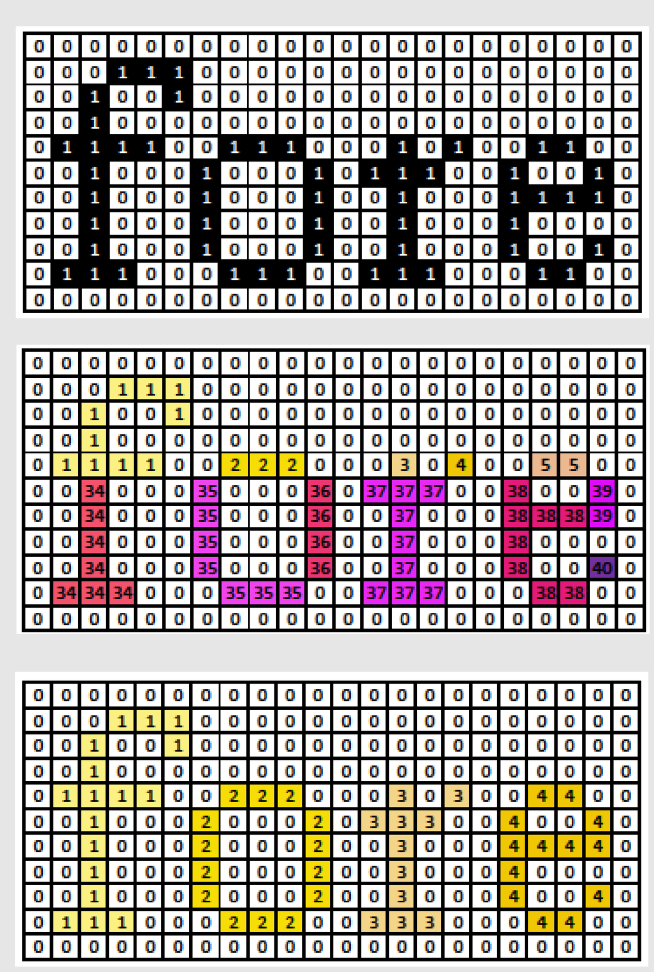


BBDT

	BBDT	BBDT_ALL	BBDT_ONE
3DPeS	0.678	**0.621**	0.645
Fingerprints	0.343	0.322	**0.320**
Hamlet	5.284	**5.048**	5.252
Medical	2.273	**2.154**	2.213
MIRflickr	0.450	**0.414**	0.424
Tobacco800	7.752	**7.283**	7.431

Results in ms on Windows PC with Intel Core i5-6600 @ 3.30 GHz and Microsoft Visual Studio 2013 (lower is better).

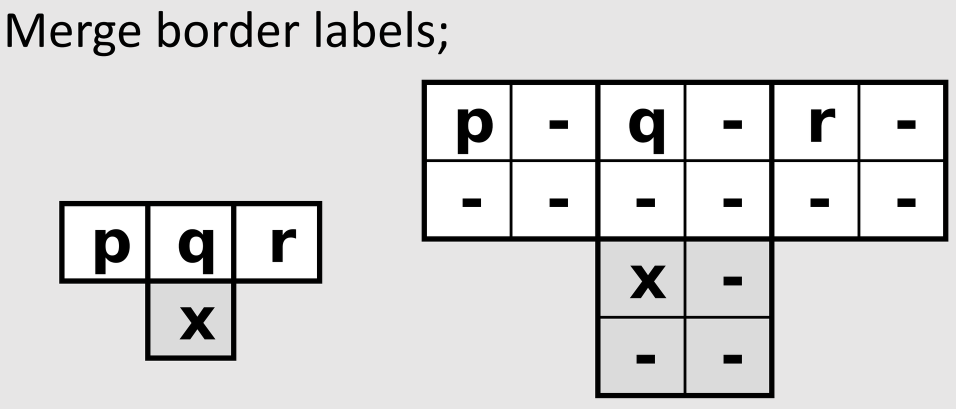
Parallel Connected Components Labeling



Idea: Improving the performance of existing algorithms employing parallelization.

Solution proposal:

- ✓ Divide image into stripes;
- ✓ Compute first scan on each stripe (in parallel);
- ✓ Merge border labels;



- ✓ Compute second scan (in parallel);

Results show that the speed up obtained with two threads on SAUF is ×1.5 in average and it increases up to ×4 on random dataset when 12 threads are involved. BBDT shows a greater speed up with low number of threads (i.e. ×1.7 with 2 threads, up to ×4.7 on random dataset with 8 threads)

	SAUF	SAUF_2	SAUF_4	SAUF_8	SAUF_12	SAUF_16	SAUF_24
3DPeS	1.817	1.258	1.013	0.886	**0.845**	0.972	0.969
Fingerprints	0.793	0.548	0.431	0.361	**0.338**	0.402	0.426
Hamlet	13.449	8.682	6.726	5.651	**5.338**	5.615	5.446
Medical	6.316	4.414	3.104	2.626	**2.542**	2.742	2.745
MIRflickr	1.053	0.770	0.608	0.544	**0.543**	0.618	0.657
Tobacco800	22.075	14.362	11.083	9.445	**9.057**	9.457	9.166
Random	31.525	19.554	11.956	8.695	**7.795**	9.144	8.605

	BBDT	BBDT_2	BBDT_4	BBDT_8	BBDT_12	BBDT_16	BBDT_24
3DPeS	1.274	0.794	**0.609**	0.720	0.812	0.918	1.002
Fingerprints	0.606	0.386	0.298	**0.274**	0.290	0.347	0.388
Hamlet	10.528	5.989	**4.342**	4.528	5.186	5.423	6.146
Medical	4.945	3.051	**1.831**	2.060	2.407	2.621	2.998
MIRflickr	0.790	0.520	**0.393**	0.438	0.479	0.559	0.622
Tobacco800	16.587	9.590	**6.518**	7.207	8.206	9.215	10.375
Random	25.106	13.177	7.084	**5.375**	6.004	7.567	8.387

Average results in ms on a virtual Windows workstation with two Intel Xeon X5650 CPU @ 2.67GHz (6 physical cores and 12 logical processors per socket) and Microsoft Visual Studio 2013 (lower is better).