TECHNISCHE
UNIVERSITÄT
WIEN

Hashcode 2019

VO: Effiziente Programme

Zachmann, Funk, Merckel
January 23, 2020

Outline

Einleitung

Der Google Hashcode

- Programmierwettbewerb von Google für Teams (max. 4)
- Befasst sich mit der Lösung von *NP* – *hard* Problemen.
- Erlaubt sind alle Programmiersprachen.
- Teamwork ist wichtig,

`https://codingcompetitions.withgoogle.com/hashcode`

Problemstellung

Hashcode Qualifikation 2019

- Erstellung von Photocollage
- Gegeben sind Beschreibungen von hypothetischen Photos
- Orientierung [h,v]
- tags [beliebige chars]

Problemstellung Forts.

Hashcode Qualifikation 2019

- Input ist gegeben als text
- Punkte werden gegeben durch ”interest factor”
- ”interest factor”: minimum von (gleich, unterschiedlich)
- Vertikale Photos werden zusammengefasst

Input

Hashcode Qualifikation 2019

- Gegeben sind fünf Textdateien
- Unterscheiden sich durch Größe und Zusammensetzung
- Größe: 4, 80000, 1000, 90000, 80000

Format:

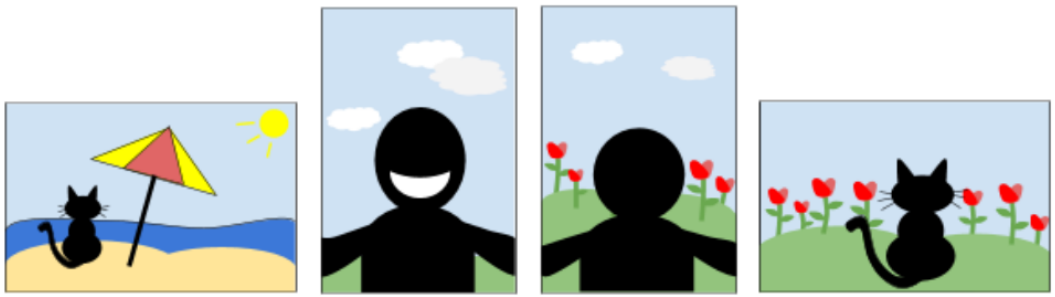
```
4
H   3   cat beach sun
```

Input Forts.

Hashcode Qualifikation 2019

Beispiel score

Example



cat, beach, sun *selfie, smile* *garden, selfie* *garden, cat*

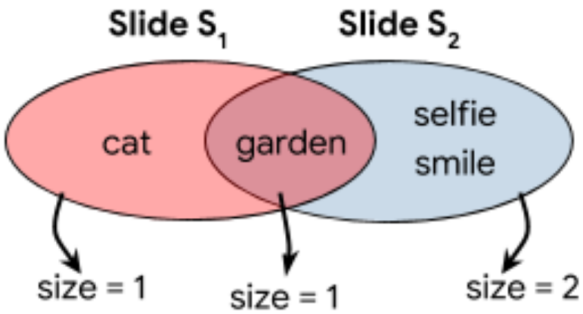
Input file	Description
4 H 3 cat beach sun V 2 selfie smile V 2 garden selfie H 2 garden cat	The collection has 4 photos Photo 0 is horizontal and has tags [cat, beach, sun] Photo 1 is vertical and has tags [selfie, smile] Photo 2 is vertical and has tags [garden, selfie] Photo 3 is horizontal and has tags [garden, cat]

Scoring

Hashcode Qualifikation 2019

Beispiel

For **example**, for the slide transition from S_1 to S_2, we know that the tags are [garden, cat] for S_1, and [selfie, smile, garden] for S_2:



Interest factor = min(1, 1, 2) = 1

- The number of common tags is 1 ⟶ [garden]
- The number of tags in S_1, but not is S_2 is 1 ⟶ [cat]
- The number of tags in S_2, but not in S_1, is 2 ⟶ [selfie and smile]

The interest factor is the minimum of these numbers, so it is 1.

Motivation

Hashcode Qualifikation 2019

- Passt gut auf VO - Inhalt
- ”Real world” - Problemstellung
- Besonders interessant wegen O und score - Opt.
- Fun!

Naiver Ansatz - greedy

Hashcode Qualifikation 2019

- Java
- geht von Photo zu Photo
- Keine Punktoptimierung
- Erstellt Liste

Proof of concept - Python

Hashcode Qualifikation 2019

- Python für prototyping
- Reduktion auf max hamiltonian trail
- Ansatz mit Matrix
- Ansatz ohne Matrix

Algo - Effizienzsteigerung: Python

Hashcode Qualifikation 2019

- max-TSP
- built-in set
- monte-carlo
- priority queue

Algo - Effizienzsteigerung Forts.

Hashcode Qualifikation 2019

- graph-symmetry
- Revisited max_tsp
- priority on demand
- discard graph

Python - Analyse

Hashcode Qualifikation 2019

c_memorable_moments.txt

	max-TSP	built-in set	monte-carlo
Laufzeit:	300.949	293.901	1.323
Punkte:	603	561	1432

Zeitlicher Aufwand: 7 ph

Python - Analyse Forts.

Hashcode Qualifikation 2019

c_memorable_moments.txt

	priority queue	graph-symmetry	Revisited
Laufzeit:	13.467	0.789	116.297
Punkte:	1494	1430	1438

Zeitlicher Aufwand: + 4 ph

Python - Analyse Forts.

Hashcode Qualifikation 2019

c_memorable_moments.txt

	priority on demand	discard graph
Laufzeit:	1.227	0.539
Punkte:	1508	1437

Zeitlicher Aufwand: + 4 ph

Effizienzsteigerung

Hashcode Qualifikation 2019

- Wechsel von Python auf C
- Zunächst 1 - 1
- Yale - Matrix
- Probleme

Low Level Effizienzsteigerung

Hashcode Qualifikation 2019

- Cycles zwischen $5 * 10^9$ (1000) und $1.2 * 10^{11}$ (5000)
- Visited array durch swapping
- ca. 10 Prozent Verbesserung in cycles und instructions

Vergleich Punkte

Hashcode Qualifikation 2019

Punktvergleiche

	a	c	f	g	h	i	j
naiv	2	158	2509	5523	9382	12437	16916
pyth	2	1508	3350	7369	12454	16607	22634
C	2	1426	3351	7363	12453	16605	22636

Vergleich Laufzeiten

Hashcode Qualifikation 2019

Laufzeitvergleiche

Ansatz	algorithmische Laufzeit
naiv	$O(n)$
python	$O(n^2)$
C	$O(n^2)$

Vergleich Effizienzen

Hashcode Qualifikation 2019

Effizienzvergleiche cycles

	a	c	f	g	h	i	j
solv.c	0.4M	575M	1379M	10355M	33095M	67566M	123060M
solv2.c	0.4M	341M	246M	1027M	2947M	5764M	10521M
solv3.c	0.4M	314M	236M	967M	2810M	5658M	10450M

Vergleich Effizienzen Forts.

Hashcode Qualifikation 2019

Effizienzvergleiche instructions

	a	c	f	g	h	i	j
solv.c	0.2M	560M	1610M	11562M	36918M	86166M	164087M
solv2.c	0.2M	253M	189M	771M	1877M	3314M	5554M
solv3.c	0.2M	238M	175M	718M	1756M	3100M	5209M

Vergleich Effizienzen Forts.

Hashcode Qualifikation 2019

Effizienzvergleiche flags (cycles)

c	o1-dbg	o1	of-dbg	of-ndbg	of
solv.c	672M	690M	520M	568M	575M
solv2.c	383M	372M	345M	314M	341M
solv3.c	360M	346M	314M	312M	314M

Punkte / LoC

Hashcode Qualifikation 2019

Sprachen und loc

	java	python	solv.c	solv2.c	solv3.c
loc	ca.220	ca.500	ca.500	ca.480	480

Zusammenfassung

Hashcode Qualifikation 2019

- Größe der Problemstellung
- Optimierungen hauptsächlich Algo
- Programmierung auf Effizienz - from scratch