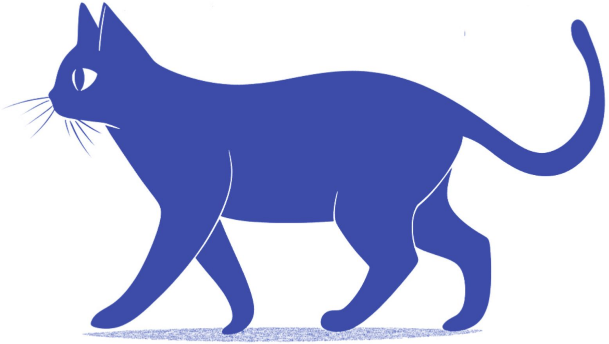
Adaptively Generating Heterogeneous Execution Strategies using the VOILA Framework

Tim Gubner



Problem: We want specific optimal Instances

Example: Book cover

- Some French word:	*VOILA / voilá*
- Sword from some British story:	*Excalibur*
- Some cat:	*Foppie*

⇨ many possible covers

Problem: We want specific optimal Instances

Example: Book cover

- Some French word: *VOILA / voilá*
- Sword from some British story: *Excalibur*
- Some cat: *Foppie*

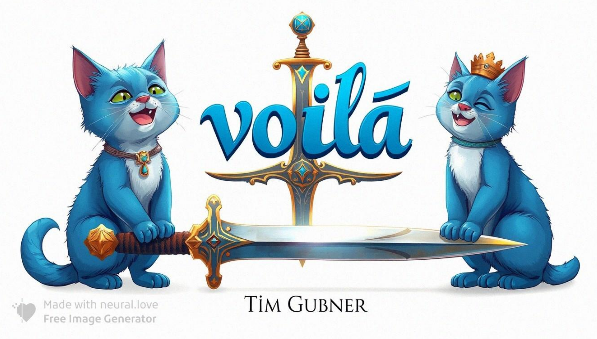
⇨ many possible covers

Often, we want the “best” instance

⇨ “best” often depends

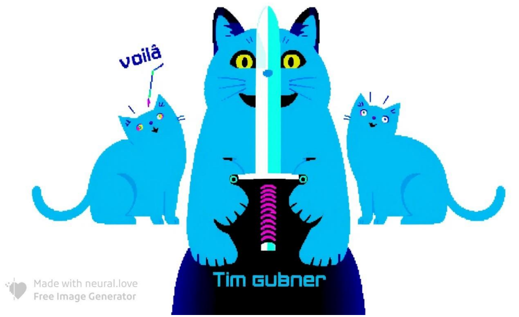
ADAPTTLINY
GENERATING
HETEROGENOUS
EXCCUIINN STRATTIVE
SREATLOES
TIM GUBNER
VOILA
Made with neural.love
Free Image Generator

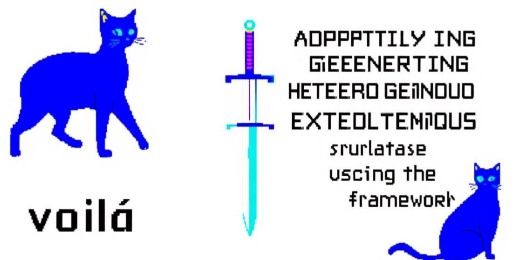
LPPIT:
LAPPT LY GENERAING
NSTT HETEROGONEOUS
EEECUENATION
STAXTEGSEFUN
ESTPALHNNEE
VOILA
FRAMEWOrR
TIM GUBNER
Made with neural.love
Free Image Generator


voilá
TIM GUBNER
Made with neural.love
Free Image Generator

voilà
?
VOILA
Th Appatively cerst HETEROGENELUS EXÉCUTION STRATEIGIES
Made with neural.love
Free Image Generator

voilã
VOILÀ
Made with neural.love
Free Image Generator


voilâ
Tim Gubner
Made with neural.love
Free Image Generator


ADPPPTTILY ING
GIEEENERTING
HETEERO GEIINOUD
EXTEDLTEMIQUS
srurlatase
uscing the
framework
voilá

voilá
adptivelly gentepeticnegaden
pxetuties using the VOila Frraworn

voilá
ADPTIVILY GENEATING HEHEROGENENEIUS STRATEGIES
USIT THE VOILA FRAMBVORK
TIMCUBBER

Application to Database Management Systems

1. User issues a query

What's the sum of the stored integers?

2. System computes result

110

Computing the Sum

Stored := 1, 1, 2, 2, 3, 3 … 10, 10 (first 10 natural integers, each twice)

“Naively” add each integer:
(1+1+2+2+3+3+4+4+5+5+6
+6+7+7+8+8+9+9+10+10) =
110

19 additions

Computing the Sum

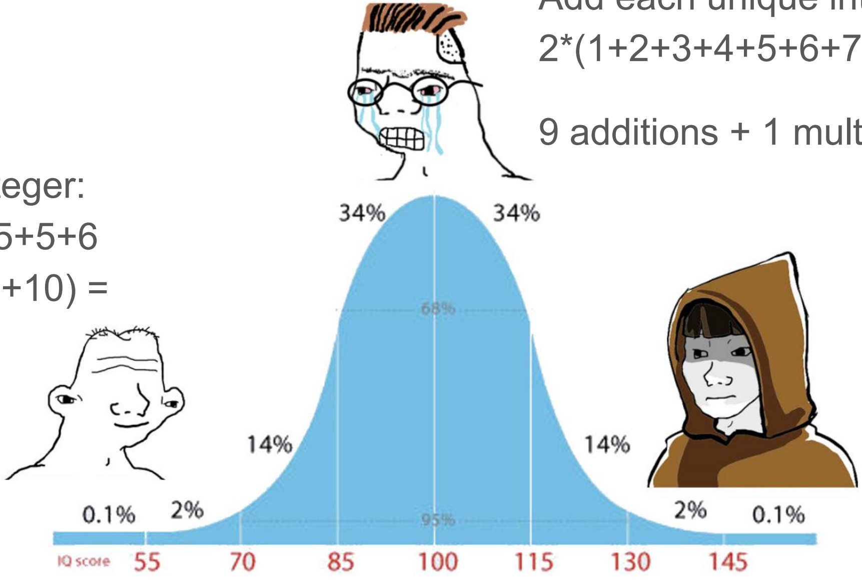
Stored := 1, 1, 2, 2, 3, 3 … 10, 10 (first 10 natural integers, each twice)

“Naively” add each integer:
(1+1+2+2+3+3+4+4+5+5+6
+6+7+7+8+8+9+9+10+10) =
110

19 additions

Add each unique integer once, multiply by 2:
2*(1+2+3+4+5+6+7+8+9+10) = 2 * 55 = 110

9 additions + 1 multiplication



Computing the Sum

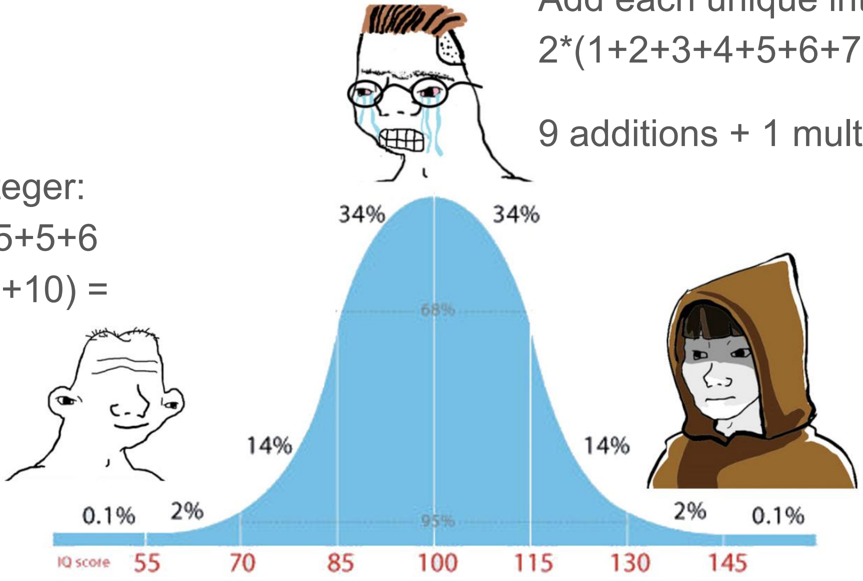
Stored := 1, 1, 2, 2, 3, 3 … 10, 10 (first 10 natural integers, each twice)

Add each unique integer once, multiply by 2:
2*(1+2+3+4+5+6+7+8+9+10) = 2 * 55 = 110

9 additions + 1 multiplication

“Naively” add each integer:
(1+1+2+2+3+3+4+4+5+5+6+6+7+7+8+8+9+9+10+10) = 110

19 additions



2 * Gauß formula: 2* (N^2 + N)/2 = N^2+N = 10 * 10 + 10 = 110

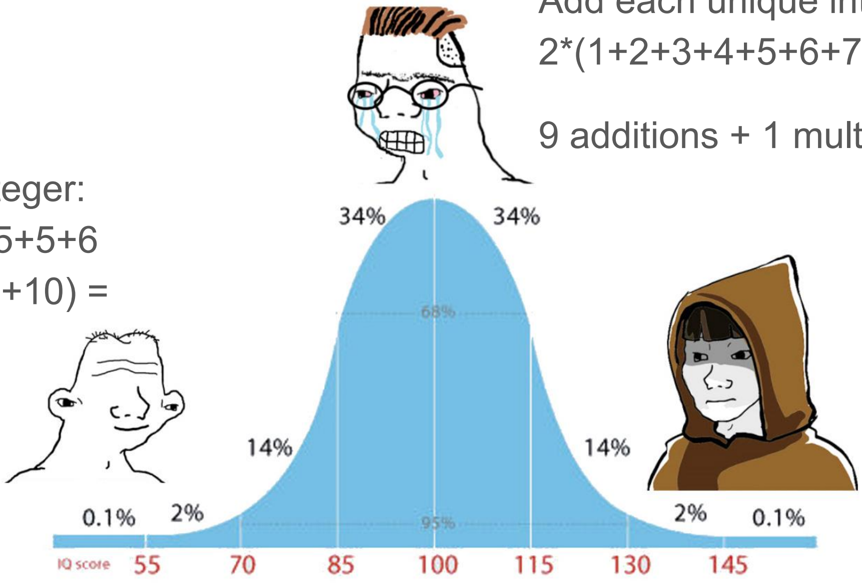
1 addition + 1 multiplication

Computing the Sum

Stored := 1, 1, 2, 2, 3, 3 … 10, 10 (first 10 natural integers, each twice)

“Naively” add each integer: (1+1+2+2+3+3+4+4+5+5+6+6+7+7+8+8+9+9+10+10) = 110

19 additions



Add each unique integer once, multiply by 2: 2*(1+2+3+4+5+6+7+8+9+10) = 2 * 55 = 110

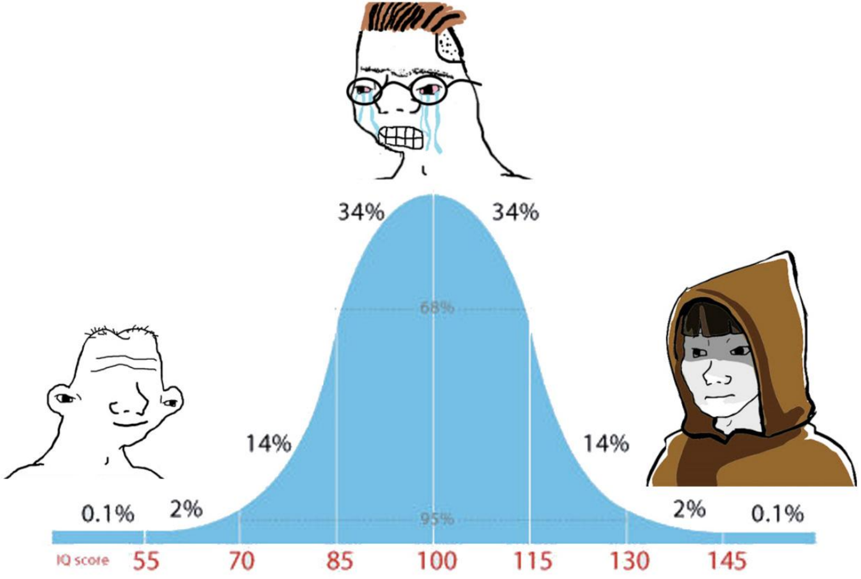
9 additions + 1 multiplication

2 * Gauß formula: 2* (N^2 + N)/2 = N^2+N = 10 * 10 + 10 = 110

1 addition + 1 multiplication

More instance-specific

Instance-Specific Optimizations



“Holy Grail”

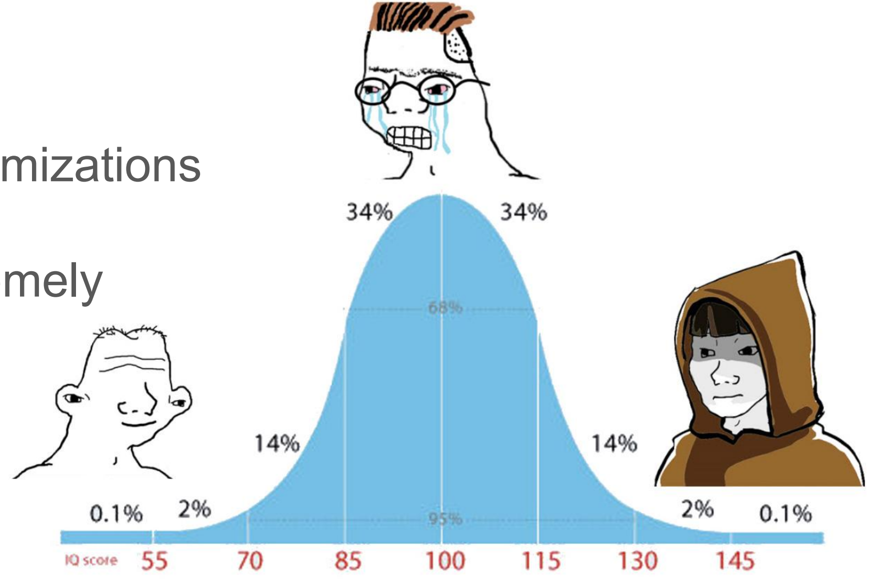
- Too specific to be practical?
- Needs to be generated by computer

More instance-specific

Instance-Specific Optimizations

“Normal” Optimizations

- Not extremely specific
- Possibly do-able



“Holy Grail”

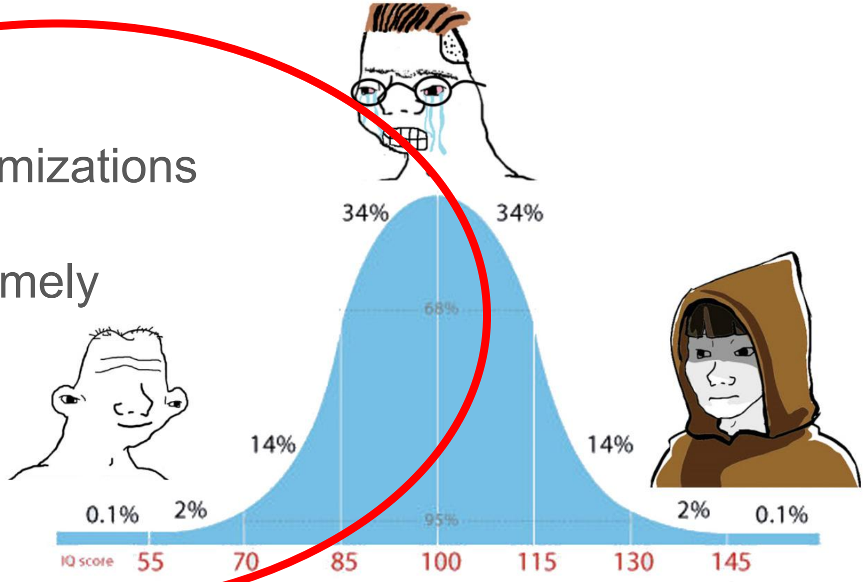
- Too specific to be practical?
- Needs to be generated by computer

More instance-specific

Instance-Specific Optimizations

“Normal” Optimizations

- Not extremely specific
- Possibly do-able



“Holy Grail”

- Too specific to be practical?
- Needs to be generated by computer

More instance-specific

Smarter = Better?

#Integers	*Naive*	*“Medium Smart”*	*“Super Smart”*
200	199 additions	99 additions, 1 multiplication	1 addition, 1 multiplication
20 (example)	19 additions	9 additions, 1 multiplication	1 addition, 1 multiplication
2	1 addition	1 multiplication	1 addition, 1 multiplication

Smarter = Better?

Assume: an addition costs 1 and multiplication 10

Costs:

#Integers	*Naive*	*“Medium Smart”*	*“Super Smart”*
200	199	109	11
20 (example)	19	19	11
2	1	10	11

Smarter = Better?

Assume: an addition costs 1 and multiplication 10

Costs:

#Integers	*Naive*	*“Medium Smart”*	*“Super Smart”*
200	199	109	**11**
20 (example)	19	19	**11**
2	**1**	10	11

Smarter = Better?

Assume: an addition costs 1 and multiplication 10

Costs:

#Integers	*Naive*	*“Medium Smart”*	*“Super Smart”*
200	199	109	**11**
20 (example)	19	19	**11**
2	**1**	10	11

- **“Smarter” not always better**
- **Runtime depends on many factors (static & dynamic)**
- **A system would need to make a choice**

This Thesis

Manual exploration:

- Data representations (e.g. number repeats)
- Algorithms optimized for specific cases
- Different implementation styles (“book cover layouts”)

VOILA: Framework to generate specific implementation styles

Excalibur: Automated exploration & adoption of better implementations, *while query is running*

voilá