Geohash in Golang Assembly

Lessons in Absurd Optimization

30 August 2018

Michael McLoughlin
Software Engineer, Uber ATG


Donald
Knuth '60
50th Reunion

What is Geohash?

Geohash is a Point Encoding Scheme

Maps a latitude longitude pair to a 64-bit integer or equivalent string.

- Indexing geo data
- Proximity queries
- Used in Redis, for example

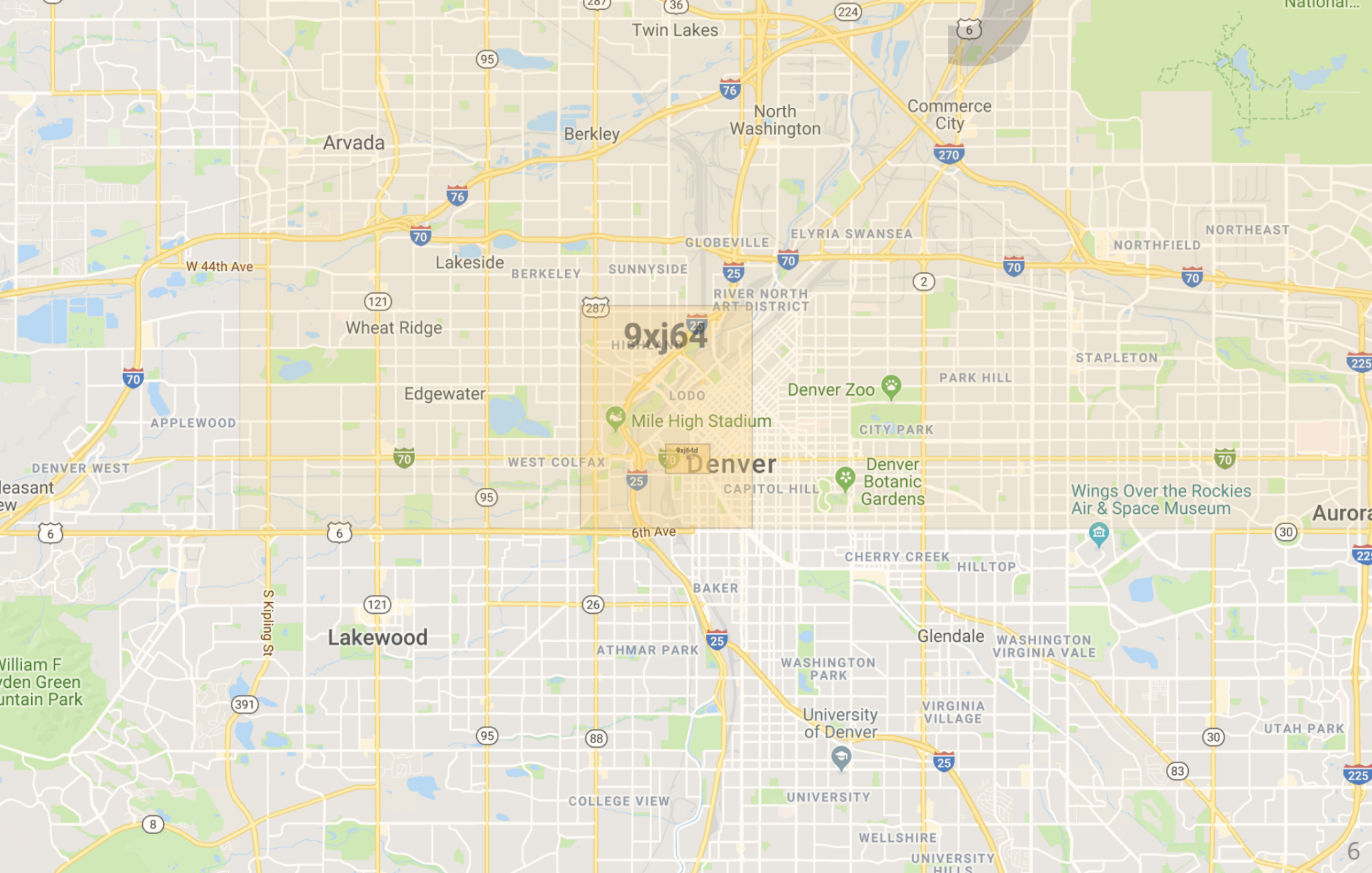
Example:

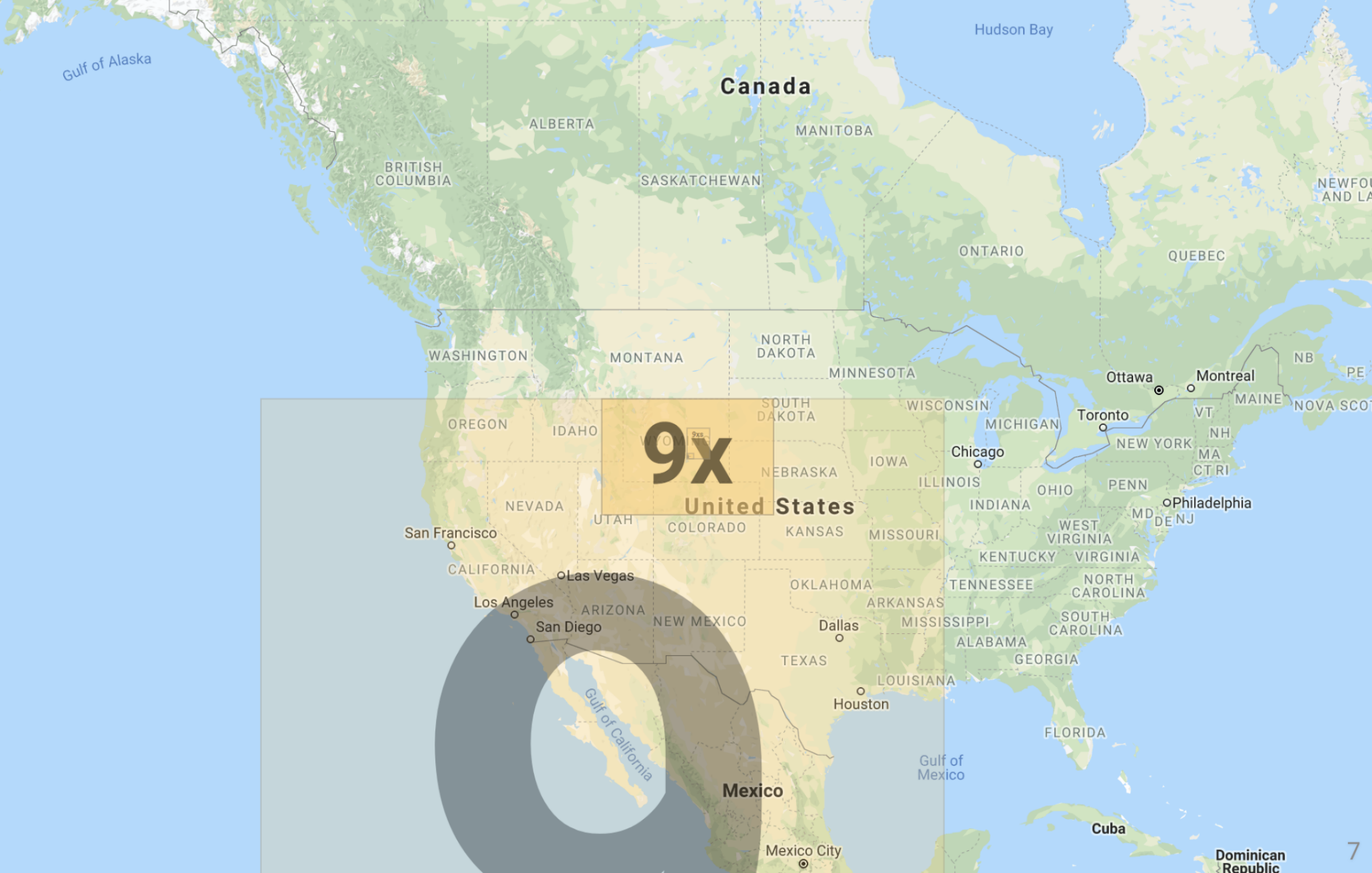
- **Gophercon** is at: `(39.74279,-104.99706)`
- Integer geohash is: `0xceb7f254240fd612`
- String geohash is: tuvz4p141zc1

Geohash can be taken at various *precisions*.

Divides the earth into *grid cells* of various sizes.


9xj64dz
9xj64dzq
Mile High Ballroom
The UPS Store


9xj64
Twin Lakes
Arvada
Berkley
North Washington
Commerce City
Lakeside
BERKELEY
SUNNYSIDE
GLOBEVILLE
ELYRIA SWANSEA
NORTHEAST
NORTHFIELD
W 44th Ave
RIVER NORTH ART DISTRICT
Wheat Ridge
Edgewater
LODO
Mile High Stadium
Denver Zoo
PARK HILL
STAPLETON
APPLEWOOD
CITY PARK
DENVER WEST
WEST COLFAX
Denver
CAPITOL HILL
Denver Botanic Gardens
Wings Over the Rockies Air & Space Museum
Aurora
6th Ave
CHERRY CREEK
HILLTOP
BAKER
Lakewood
S Kipling St
ATHMAR PARK
Glendale
WASHINGTON VIRGINIA VALE
WASHINGTON PARK
VIRGINIA VILLAGE
University of Denver
UTAH PARK
COLLEGE VIEW
UNIVERSITY
WELLSHIRE
UNIVERSITY HILLS


Gulf of Alaska
Canada
Hudson Bay
ALBERTA
MANITOBA
BRITISH COLUMBIA
SASKATCHEWAN
ONTARIO
QUEBEC
NEWFOUNDLAND AND LABRADOR
WASHINGTON
MONTANA
NORTH DAKOTA
MINNESOTA
NB
PE
Ottawa
Montreal
OREGON
IDAHO
SOUTH DAKOTA
WISCONSIN
MICHIGAN
Toronto
MAINE
NOVA SCOTIA
VT
NH
NEW YORK
MA
CT RI
9x
United States
NEBRASKA
IOWA
Chicago
ILLINOIS
OHIO
PENN
Philadelphia
NEVADA
UTAH
COLORADO
KANSAS
MISSOURI
INDIANA
MD
DE
NJ
WEST VIRGINIA
San Francisco
KENTUCKY
VIRGINIA
CALIFORNIA
Las Vegas
OKLAHOMA
TENNESSEE
NORTH CAROLINA
Los Angeles
ARIZONA
ARKANSAS
San Diego
NEW MEXICO
Dallas
MISSISSIPPI
SOUTH CAROLINA
ALABAMA
TEXAS
GEORGIA
LOUISIANA
Houston
Gulf of California
FLORIDA
Gulf of Mexico
Mexico
Cuba
Mexico City
Dominican Republic

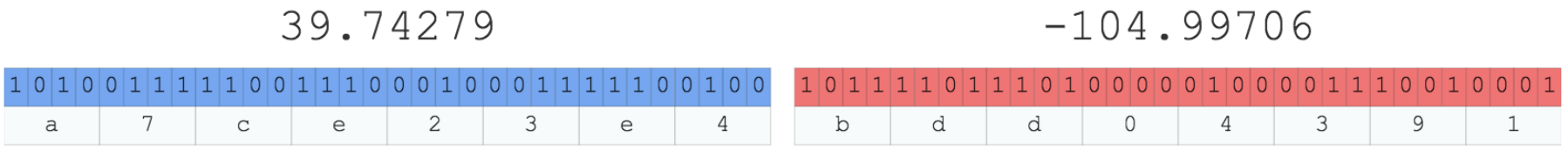

Greenland
Iceland
Finland
Sweden
Norway
United Kingdom
Poland
Germany
Ukraine
France
Italy
Spain
Turkey
Russia
Canada
Mongolia
China
South Korea
Japan
India
Thailand
United States
9
9s
North Pacific Ocean
North Atlantic Ocean
Algeria
Libya
Egypt
Saudi Arabia
Iraq
Iran
Mali
Niger
Chad
Sudan
Nigeria
Ethiopia
Kenya
DRC
Tanzania
Angola
Namibia
Botswana
Madagascar
South Africa
Venezuela
Colombia
Peru
Brazil
Bolivia
Chile
Argentina
Indonesia
Papua New Guinea
Australia
New Zealand
Indian Ocean
South Pacific Ocean
South Atlantic Ocean
Southern Ocean

Forget about String Geohashes

For *internal* or performance-sensitive use cases integer form is enough.

How it Works

Step 1: Quantize latitude/longitude to 32-bits



- Input: (39.74279, -104.99706)
- Quantized: (0xa7ce23e4, 0xbdd04391)

Step 2: Bit Interleave

39.74279

1	0	1	0	0	1	1	1	1	1	0	0	1	1	1	0	0	0	1	0	0	0	1	1	1	1	1	0	0	1	0	0
a		7		c		e		2		3		e		4																	

-104.99706

1	0	1	1	1	1	0	1	1	1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1	0	0	1	0	0	0	1
b		d		d		0		4		3		9		1																	

1	1	0	0	1	1	1	0	1	0	1	1	0	1	1	1	1	1	1	1	0	0	1	0	0	1	0	1	0	1	0	0	0	0	1	0	0	1	0	1	1	1	1	1	1	0	1	0	1	1	0	0	0	1	0	0	1	0
c		e		b		7		f		2		5		4		2		4		0		f		d		6		1		2																													

- Input: (39.74279, -104.99706)
- Quantized: (0xa7ce23e4, 0xbdd04391)
- Interleaved: 0xceb7f254240fd612

This is the *integer geohash*.

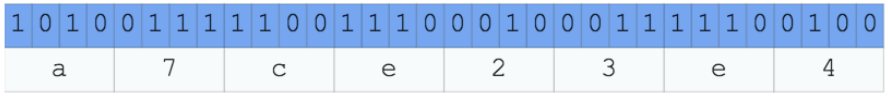
Pure Go

High Level

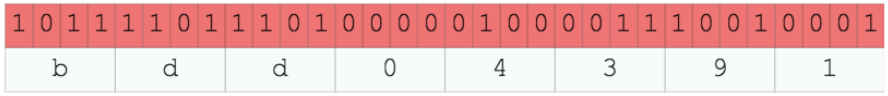
```
// EncodeInt encodes the point (lat, lng) to a 64-bit integer geohash.
func EncodeInt(lat, lng float64) uint64 {
    return Interleave(Quantize(lat, lng))
}
```

Quantize

39.74279

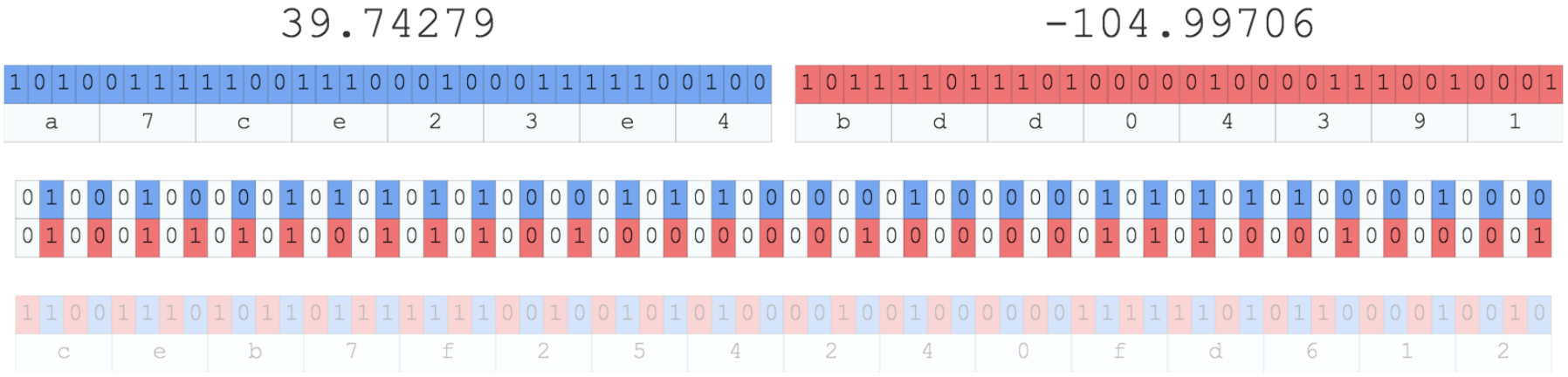


-104.99706



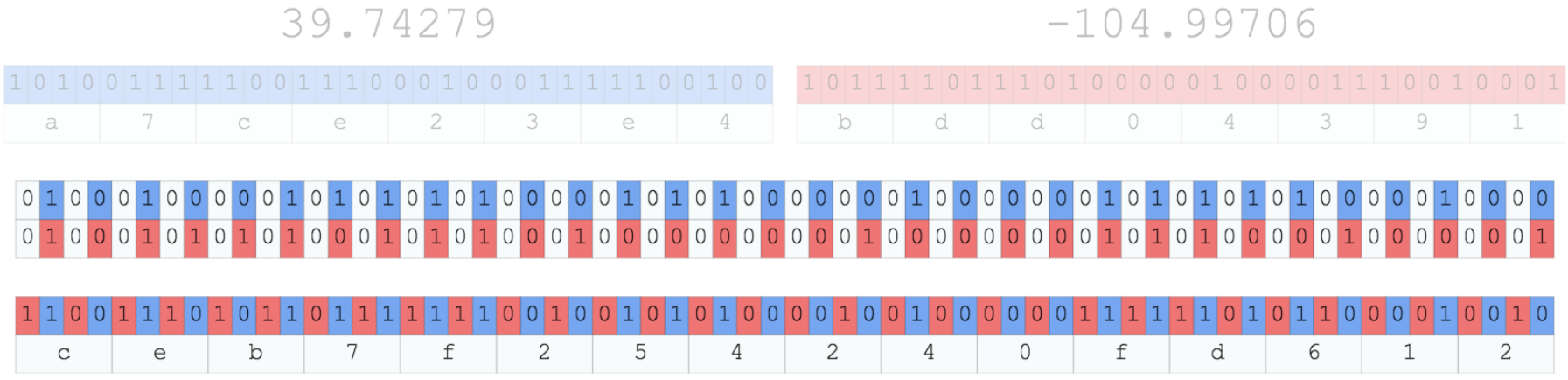

```
// Quantize maps latitude and longitude to 32-bit integers.
func Quantize(lat, lng float64) (lat32 uint32, lng32 uint32) {
    lat32 = uint32(math.Ldexp((lat+90.0)/180.0, 32))
    lng32 = uint32(math.Ldexp((lng+180.0)/360.0, 32))
    return
}
```

Intermediate Step: Spread




```
// Spread out the 32 bits of x into 64 bits, where the bits of x occupy even
// bit positions.
func Spread(x uint32) uint64 {
    X := uint64(x)
    X = (X | (X << 16)) & 0x0000ffff0000ffff
    X = (X | (X << 8)) & 0x00ff00ff00ff00ff
    X = (X | (X << 4)) & 0x0f0f0f0f0f0f0f0f
    X = (X | (X << 2)) & 0x3333333333333333
    X = (X | (X << 1)) & 0x5555555555555555
    return X
}
```

Interleave




```
// Interleave the bits of x and y. In the result, x and y occupy even and odd
// bitlevels, respectively.
func Interleave(x, y uint32) uint64 {
    return Spread(x) | (Spread(y) << 1)
}
```

27.40

Nanoseconds / Geohash

36.50

Million Geohashes / Second

Pure Go Performance

Assembly

17 Instructions

```
TEXT ·EncodeIntAsm(SB), NOSPLIT, $0
    MOVSD lat+0(FP), X0
    MOVSD lng+8(FP), X1

    MULSD $(0.005555555555555556), X0
    ADDSD $(1.5), X0
    MOVQ X1, R9
    SHRQ $20, R9

    MULSD $(0.002777777777777778), X1
    ADDSD $(1.5), X1
    MOVQ  X0, R8
    SHRQ  $20, R8

    MOVQ $0x5555555555555555, BX
    PDEPQ BX, R8, R11
    PDEPQ BX, R9, R10
    SHLQ $1, R10
    XORQ R10, R11

    MOVQ R11, ret+16(FP)
    RET
```

How to draw an owl

1.

2.

1. Draw some circles

2. Draw the rest of the owl

Scale

```
TEXT ·EncodeIntAsm(SB), NOSPLIT, $0
    MOVSD lat+0(FP), X0
    MOVSD lng+8(FP), X1

    MULSD $(0.005555555555555556), X0
    ADDSD $(1.5), X0
    MOVQ X1, R9
    SHRQ $20, R9

    MULSD $(0.0027777777777777778), X1
    ADDSD $(1.5), X1
    MOVQ  X0, R8
    SHRQ  $20, R8

    MOVQ $0x5555555555555555, BX
    PDEPQ BX, R8, R11
    PDEPQ BX, R9, R10
    SHLQ $1, R10
    XORQ R10, R11

    MOVQ R11, ret+16(FP)
    RET
```

Unit Interval Trick

```
TEXT ·EncodeIntAsm(SB), NOSPLIT, $0
    MOVSD lat+0(FP), X0
    MOVSD lng+8(FP), X1

    MULSD $(0.005555555555555556), X0
    ADDSD $(1.5), X0
    MOVQ X1, R9
    SHRQ $20, R9

    MULSD $(0.002777777777777778), X1
    ADDSD $(1.5), X1
    MOVQ  X0, R8
    SHRQ  $20, R8

    MOVQ $0x5555555555555555, BX
    PDEPQ BX, R8, R11
    PDEPQ BX, R9, R10
    SHLQ $1, R10
    XORQ R10, R11

    MOVQ R11, ret+16(FP)
    RET
```

PDEP

```
TEXT ·EncodeIntAsm(SB), NOSPLIT, $0
    MOVSD lat+0(FP), X0
    MOVSD lng+8(FP), X1

    MULSD $(0.005555555555555556), X0
    ADDSD $(1.5), X0
    MOVQ X1, R9
    SHRQ $20, R9

    MULSD $(0.002777777777777778), X1
    ADDSD $(1.5), X1
    MOVQ  X0, R8
    SHRQ  $20, R8

    MOVQ $0x5555555555555555, BX
    PDEPQ BX, R8, R11
    PDEPQ BX, R9, R10
    SHLQ $1, R10
    XORQ R10, R11

    MOVQ R11, ret+16(FP)
    RET
```

Interleave

```
TEXT ·EncodeIntAsm(SB), NOSPLIT, $0
    MOVSD lat+0(FP), X0
    MOVSD lng+8(FP), X1

    MULSD $(0.005555555555555556), X0
    ADDSD $(1.5), X0
    MOVQ X1, R9
    SHRQ $20, R9

    MULSD $(0.002777777777777778), X1
    ADDSD $(1.5), X1
    MOVQ  X0, R8
    SHRQ  $20, R8

    MOVQ $0x5555555555555555, BX
    PDEPQ BX, R8, R11
    PDEPQ BX, R9, R10
    SHLQ $1, R10
    XORQ R10, R11

    MOVQ R11, ret+16(FP)
    RET
```

2.32

Nanoseconds / Geohash

431.03

Million Geohashes / Second

Assembly Implementation

SIMD

- What if we use special instructions to do **four geohashes at once**
- Intel Intrisics
- Interleaving trick thanks to Daniel Lemire

```
TEXT ·EncodeIntSimd(SB), NOSPLIT, $0
    MOVQ lat+0(FP), AX
    MOVQ lng+24(FP), BX
    MOVQ hash+48(FP), CX

    VBROADCASTSD reciprocal180<>+0x00(SB), Y0
    VMULPD       (AX), Y0, Y0
    VBROADCASTSD onepointfive<>+0x00(SB), Y1
    VADDPD       Y1, Y0, Y0
    VPSRLQ       $20, Y0, Y0
    VBROADCASTSD reciprocal360<>+0x00(SB), Y2
    VMULPD       (BX), Y2, Y2
    VADDPD       Y1, Y2, Y1
    VPSRLQ       $20, Y1, Y1
    VMOVDQU      spreadbyte<>+0x00(SB), Y2
    VPSHUFB      Y2, Y0, Y0
    VBROADCASTSD lonibblemask<>+0x00(SB), Y3
    VPAND        Y3, Y0, Y4
    VMOVDQU      spreadnibblelut<>+0x00(SB), Y5
```

1.13	883.00
Nanoseconds / Geohash	Million Geohashes / Second

SIMD Implementation

Conclusion

- Massive performance gains are possible
- Generally prefer to rely on the compiler
- Assembly may make sense to access of special instruction sets

Full details:

- Geohash in Golang Assembly blog post
- mmcloughlin/geohash package
- Comparison of all Golang geohash packages

Thank you

Michael McLoughlin
Software Engineer, Uber ATG
mmcloughlin@gmail.com
https://mmcloughlin.com/
@mbmcloughlin