Lecture 12:

Succinct Data Structures (ctd.)
Distance Oracles

Johannes Fischer

Sparse Bit Vectors

- remainder from last lecture:
- Bit-vector $B[0,n\text{-}1]$ containing u 1's can be stored in $O(u \lg(n/u))+o(n)$ bits such that
  - $B[i]$: access to i'th bit
  - $rank_1(B,i)$: # 1's in $B[0,i]$
  - $select_1(B,i)$: pos. of i'th 1 in B
- all in O(1) time
- pioneers: $u=n/\lg n \Rightarrow O(n \lg\lg n/\lg n)=o(n)$

Simplification

- Already have o(n)-bit DS for rank/select
  - assumes we can access a **chunk** of lg n consecutive bits in B in O(1) time (table lookup!)
  - **strategy**: store B in compressed form such that C=O(lg n) consecutive bits

    $B[i..i+C-1]$

    can be read in O(1) time
  - combine with o(n) DS for "normal" rank/select

Compressing *B*

- Divide *B* into **blocks** $B_0,\ldots,B_{n/s}$ of size $s = \lg n/2$ (again!)
- For every block B_i store:
  - u_i : # 1's in B_i
  - o_i : **index** in an **enumeration** of all length-s bit-vectors with u_i 1's
- **Universal table** BlkContents[·][·]:
  - BlkContents[u][o] = bit-pattern describing block
  - |BlkContents| = $2^s \cdot s \times s = o(n)$ bits (as usual!)

Compressing *B*

- u_i's: stored in an **array** $U[0,n/s]$
  - $|U| = O(n/s \lg s) = O(n \lg\lg n/\lg n)$ bits
- o_i's: stored as $O = o_0 o_1 \ldots o_{n/s}$ (variable lengths!)
- recovering o_i from O:
  - group s blocks into a **superblock**
  - Sblk[i]: start of i'th superblock in O
    - $|\text{Sblk}| = O(n/s^2\ s) = O(n/\lg n)$ bits
  - Blk[i]: start of i'th block, **relative** to superblock
    - $|\text{Blk}| = O(n/s \lg s) = O(n \lg\lg n/\lg n)$ bits

Size of O

- important fact:

$$\sum_{i=0}^{n/s} \lg \binom{s}{u_i} \leq \lg \binom{n}{u}$$

since

$$\binom{n}{u} \geq \binom{s}{u_0} \times \cdots \times \binom{s}{u_{n/s}}$$

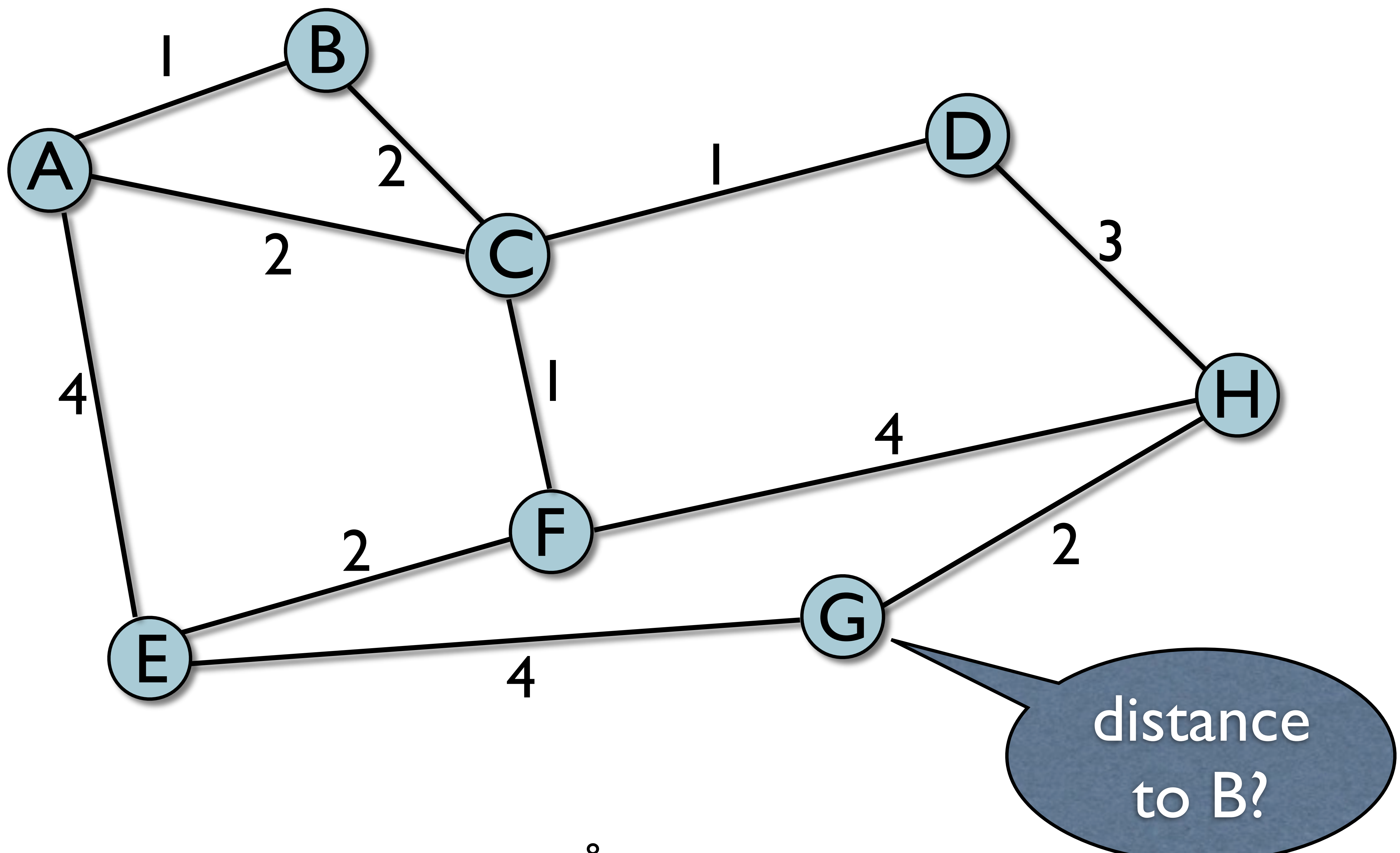
- rest of analysis: see blackboard

$\Longrightarrow$ $|O| = O(u \lg(n/u) + n/\lg n)$ bits

Distance Oracles in Graphs

- M. Thorup, U. Zwick: *Approximate Distance Oracles*. J. ACM **52**(1), 2005.

Distance Oracles



Basic Definitions

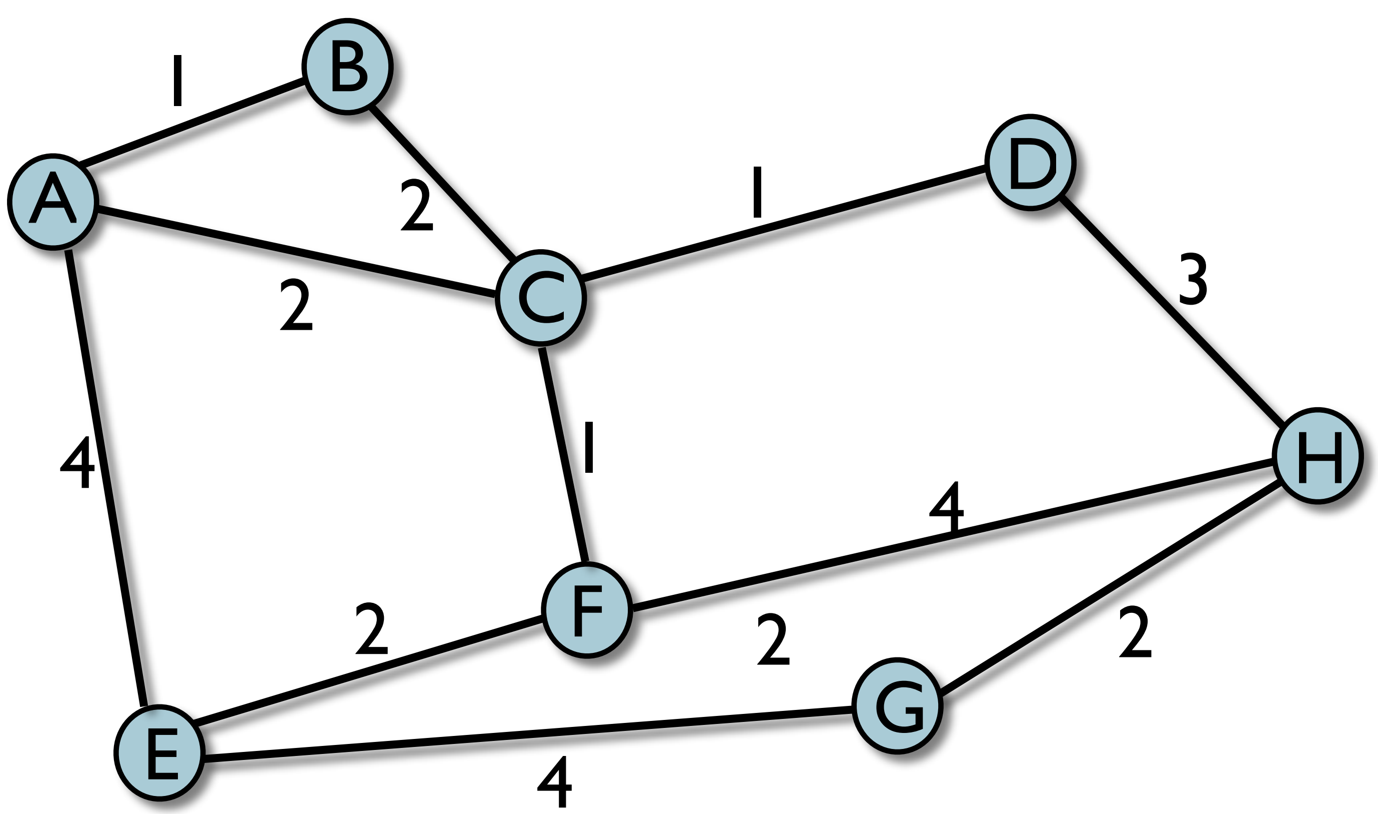
- $G=(V, E)$: weighted undirected **graph**
  - $|V|=n$, $|E| = m$
  - $e \in E$: $\omega(e) \geq 0$ **edge weights**
  - $\delta(u,v)$: **distance** from u to v
    (length of shortest path)
  - $\delta(A,v)$: **distance** from v to nearest $a \in A \subseteq V$
- $d(u,v)$ **stretch**-t approximation to $\delta(u,v)$
  $\Leftrightarrow \delta(u,v) \leq d(u,v) \leq t \cdot \delta(u,v)$

Main Result

- k: arbitrary parameter
- Preprocess G in $O(kn^{1/k}(n \lg n + m))$ time
  - DS of size $O(kn^{1+1/k})$ (**words** from now on)
  - distance **queries** $\mathrm{dist}_k(u,v)$ in $O(k)$ time
  - stretch $\leq 2k-1$
  - report **path** of length $\leq \mathrm{dist}_k(u,v)$ in $O(1)$ time per edge
- e.g. $k=2$
  - $O(n^{3/2})$ space, $O(1)$ query time, stretch 3

Part I: Metric Spaces

- Assume metric as distance matrix



⇒

	A	B	C	D	E	F	G	H
A	0	1	2	3	4	3	8	6
B		0	2	3	5	3	8	6
C			0	1	3	1	6	4
D				0	4	2	5	3
E					0	2	4	6
F						0	6	4
G							0	2
H								0

Random Samples

- construct **randomly**

  $V = A_0 \supseteq A_1 \supseteq \ldots \supseteq A_{k-1} \supseteq A_k = \emptyset$

- **Rule:**

  "Place $v \in A_{i-1}$ in A_i, independently, with probability $n^{-1/k}$."

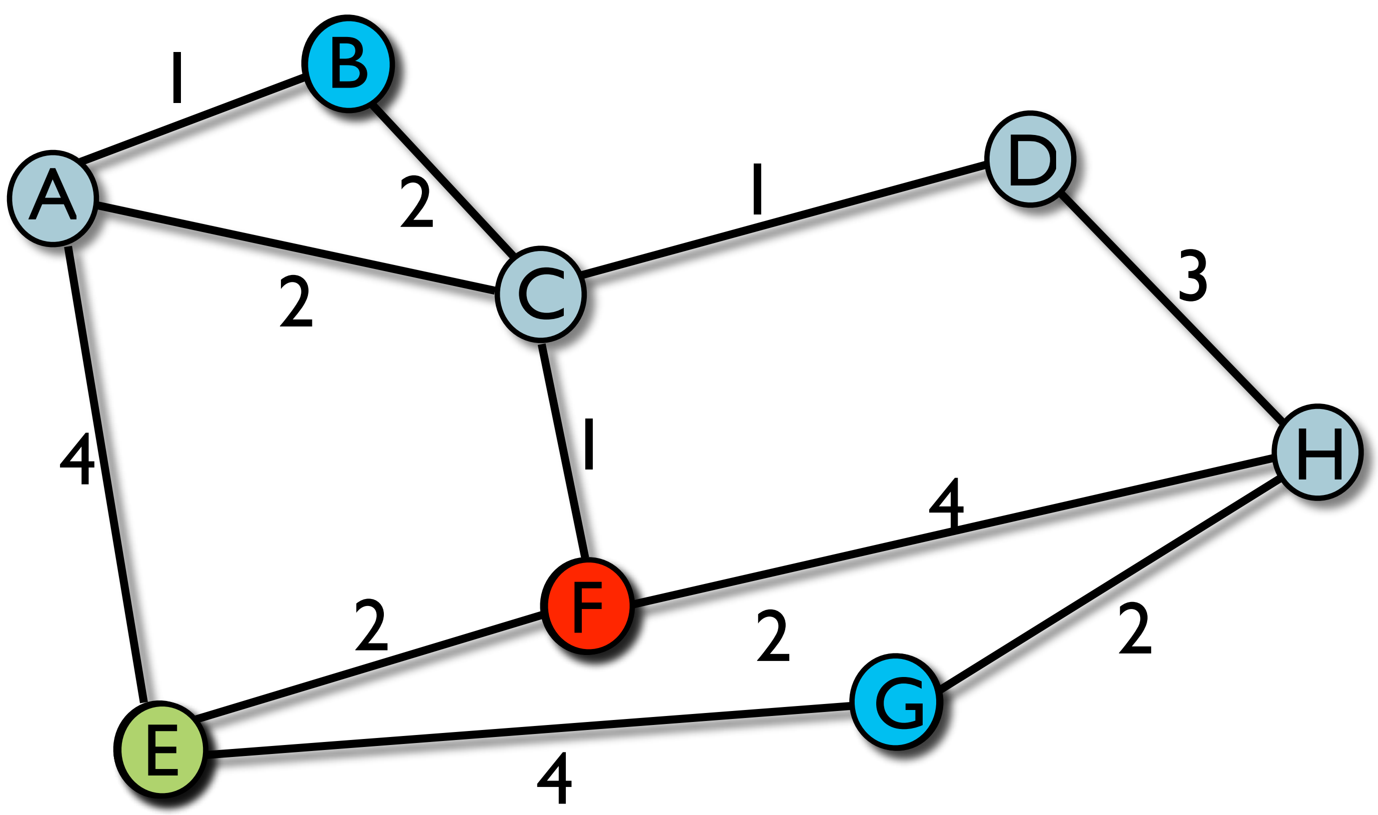
$\Longrightarrow$ $\text{Exp}[|A_i|] = |V| \times \text{Prob}[v \in A_j \ \forall\ 1 \leq j \leq i]$

$= n \times n^{-1/k} \cdot n^{-1/k} \cdot \ldots \cdot n^{-1/k}$ (i times)

$= n^{1-i/k}$

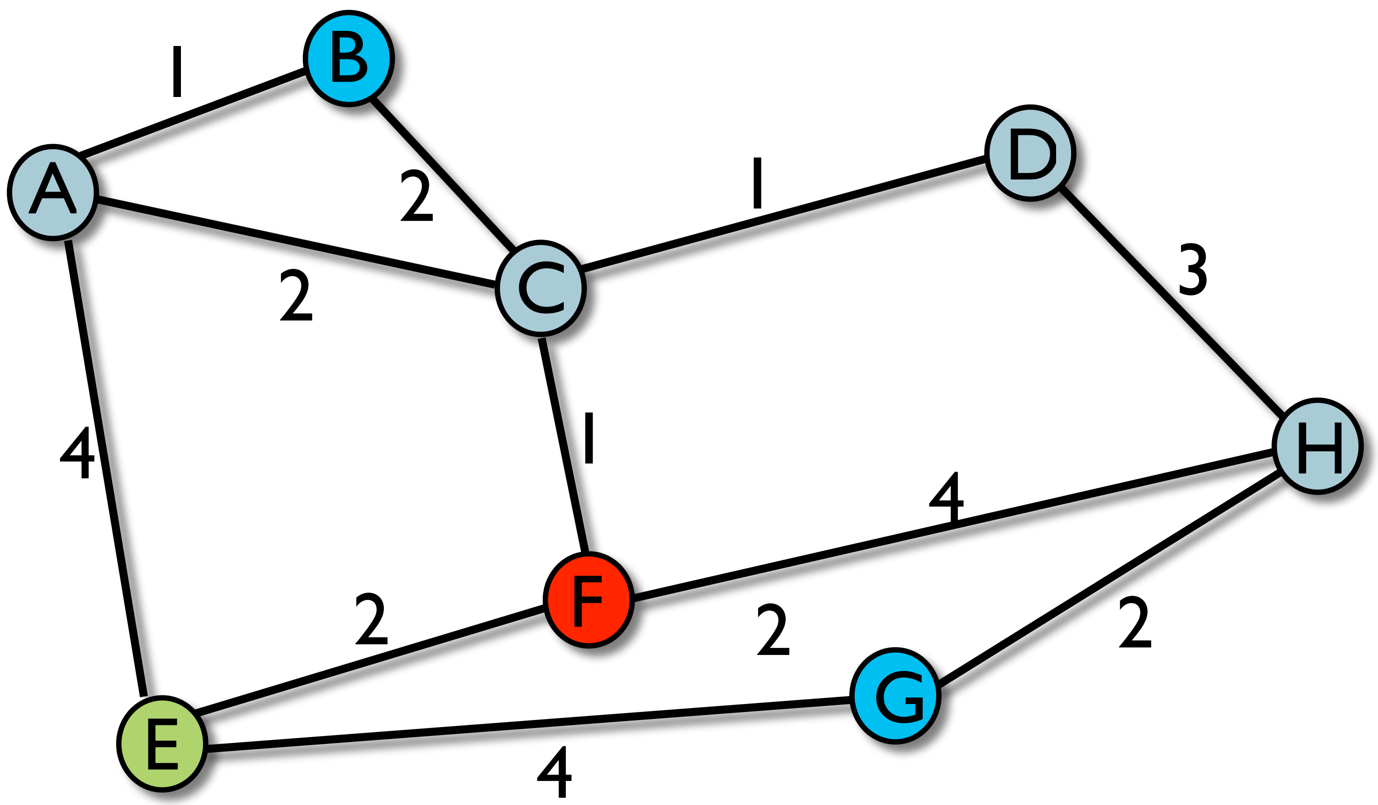
- **compute and store** $\forall v, \forall k$:
  - $\delta(A_i,v)$ and $p_i(v)$ with $\delta(p_i(v),v)=\delta(A_i,v)$ ("witness")

Example



- A_0 = {A,B,C,D,E,F,G,H}
- A_1 = {B,E,F,G}
- A_2 = {E,F}
- A_3 = {E}
- $A_4 = \varnothing$

Example

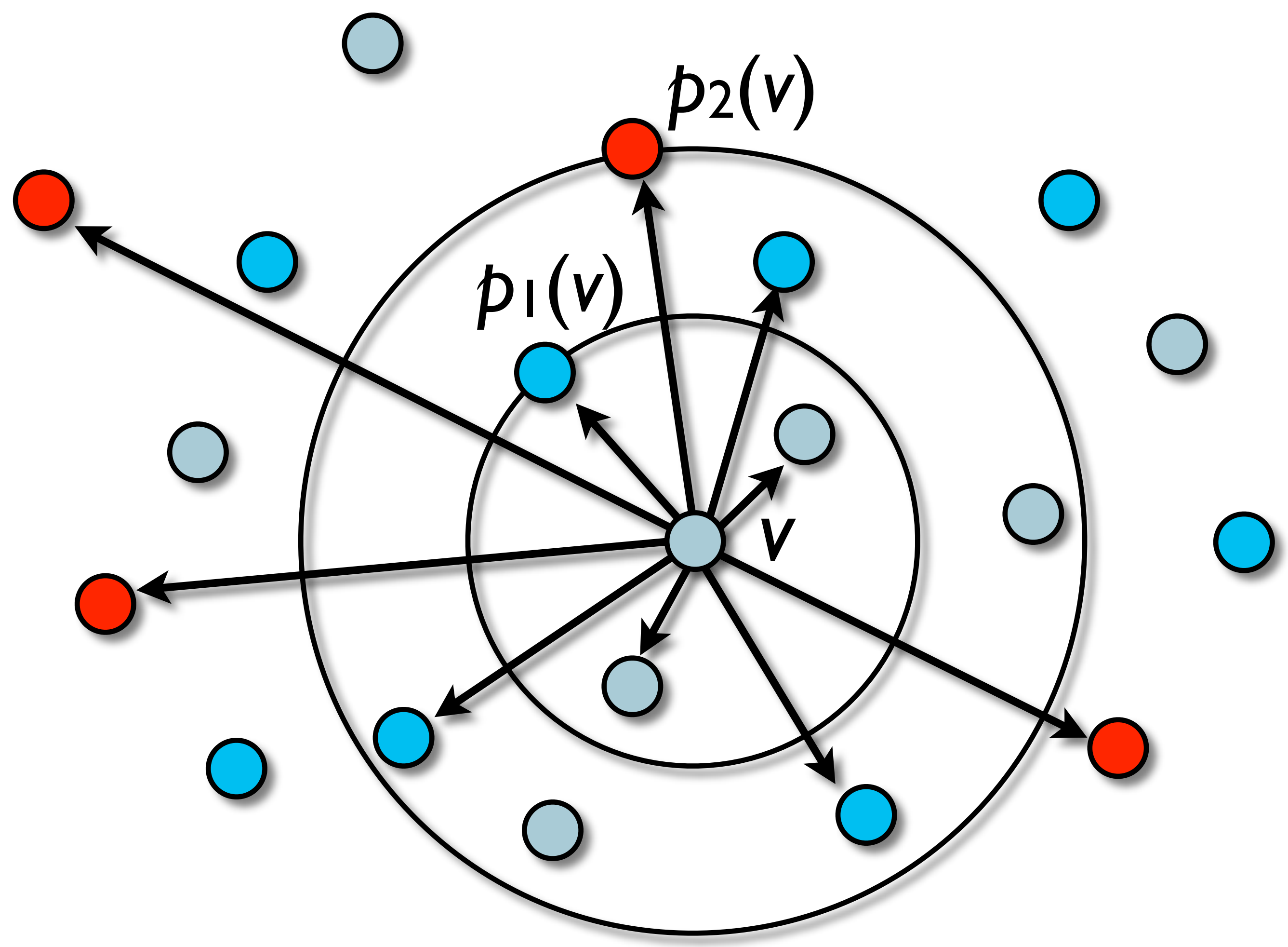


	$\delta(A_i,v)$					$p_i(v)$				
	0	1	2	3	4	0	1	2	3	4
A	0	1	3	4	∞	A	B	F	E	⊥
B	0	0	3	5	∞	B	B	F	E	⊥
C	0	1	1	3	∞	C	F	F	E	⊥
D	0	2	2	4	∞	D	F	F	E	⊥
E	0	0	0	0	∞	E	E	E	E	⊥
F	0	0	0	2	∞	F	F	F	E	⊥
G	0	0	4	4	∞	G	G	E	E	⊥
H	0	2	4	6	∞	H	G	F	E	⊥

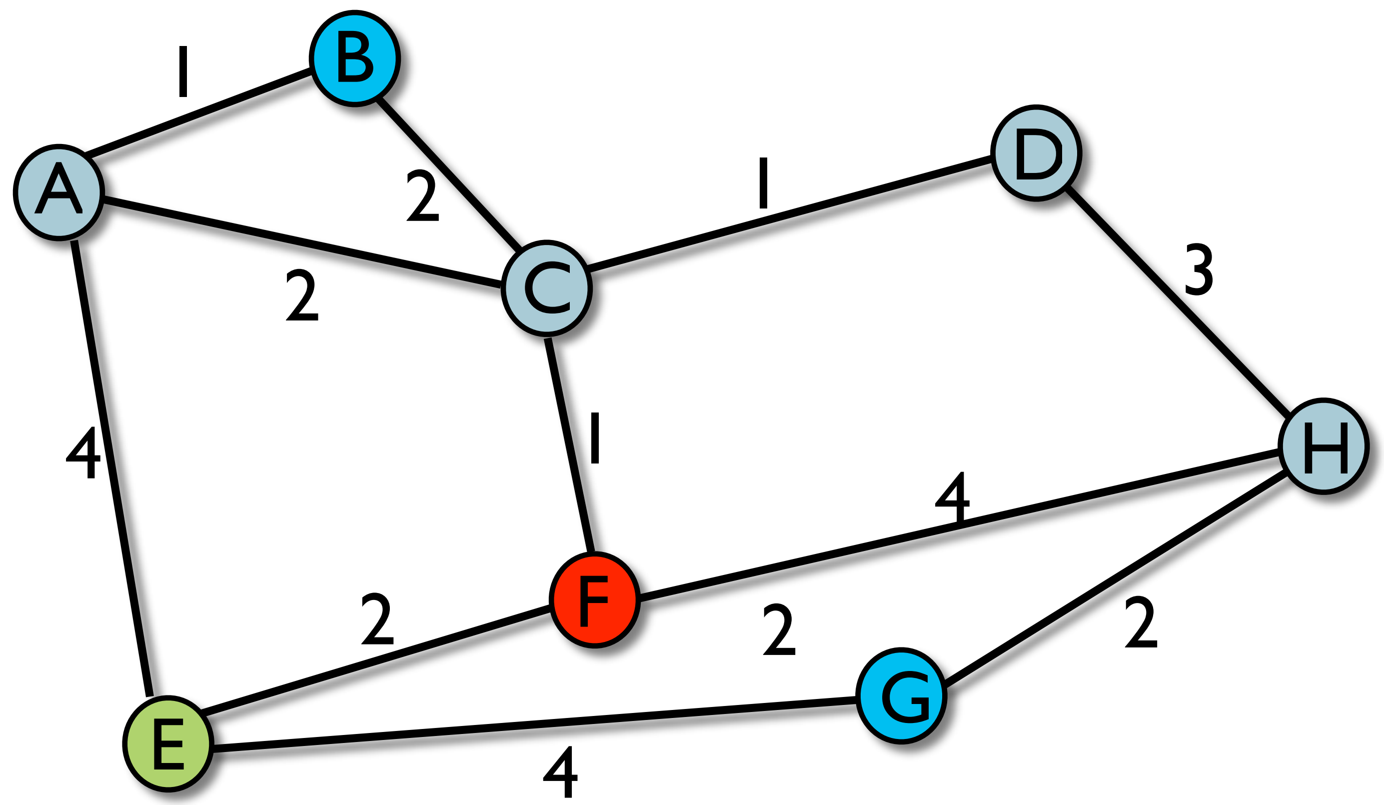
size $O(kn)$

Bunches

- **bunch** $B(v)$ of $v \in V$:
  - $w \in B(v) \Leftrightarrow \exists i : w \in A_i \setminus A_{i+1}$ and $\delta(w,v) < \delta(A_{i+1},v)$



Example



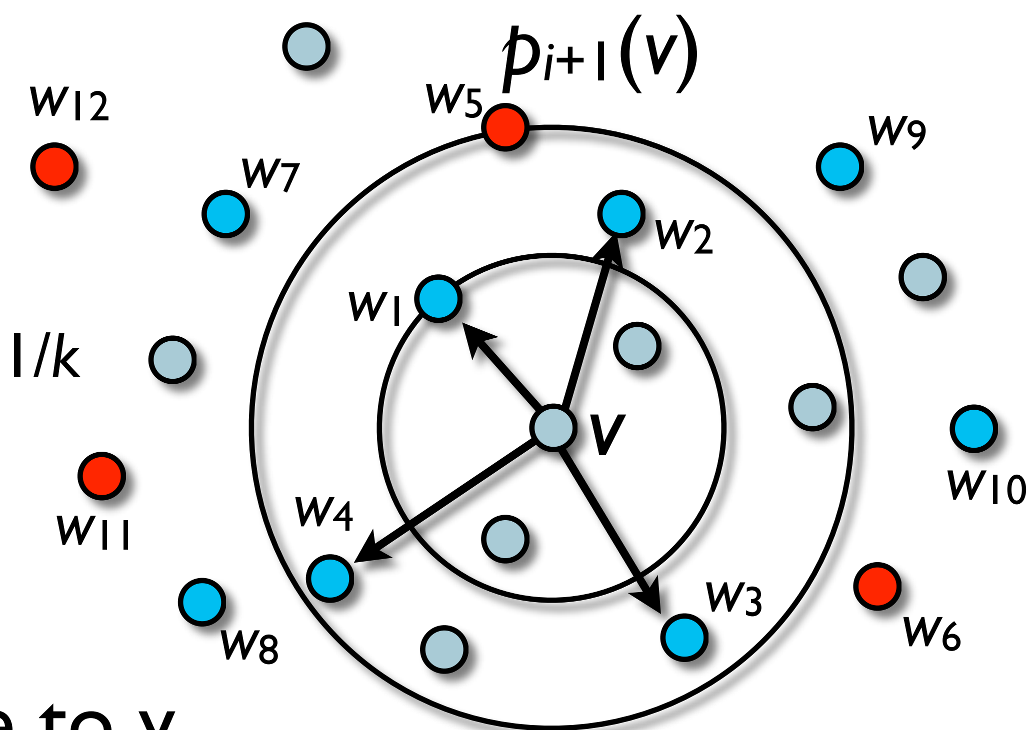
	$\delta(A_i,v)$					$p_i(v)$				
	0	1	2	3	4	0	1	2	3	4
A	0	1	3	4	∞	A	B	F	E	⊥
B	0	0	3	5	∞	B	B	F	E	⊥
C	0	1	1	3	∞	C	F	F	E	⊥
D	0	2	2	4	∞	D	F	F	E	⊥
E	0	0	0	0	∞	E	E	E	E	⊥
F	0	0	0	2	∞	F	F	F	E	⊥
G	0	0	4	4	∞	G	G	E	E	⊥
H	0	2	4	6	∞	H	G	F	E	⊥

B(A)={A, B, F, E}

1=δ(A,B) < δ(A_2,A)=3

Bunches

- Store bunches in **perfect hash table**
  - can tell in O(1) time if $w \in B(v)$...
  - ...and if so, what is $\delta(w,v)$
- $\text{Exp}[|B(v)|] \le kn^{1/k}$
  - show $\text{Exp}[|B(v) \cap (A_i \setminus A_{i+1})|] \le n^{1/k}$
  - Trivial for $i=k-1$. For $i<k-1$:
    - $w_1, w_2, \ldots, w_x$ ordered by distance to v
    - $w_j \in B(v) \Rightarrow \delta(w_j,v) < \delta(A_{i+1},v) \Rightarrow w_1,\ldots,w_{j-1} \notin A_{i+1}$
    - $\text{Prob}[w_j \in B(v)] \le (1-n^{-1/k})^j \Rightarrow \ldots \Rightarrow \text{Exp}[|B(v)|] \le kn^{1/k}$



Query Algorithm

function $\text{dist}_k(u,v)$:

$w \leftarrow u; i \leftarrow 0;$

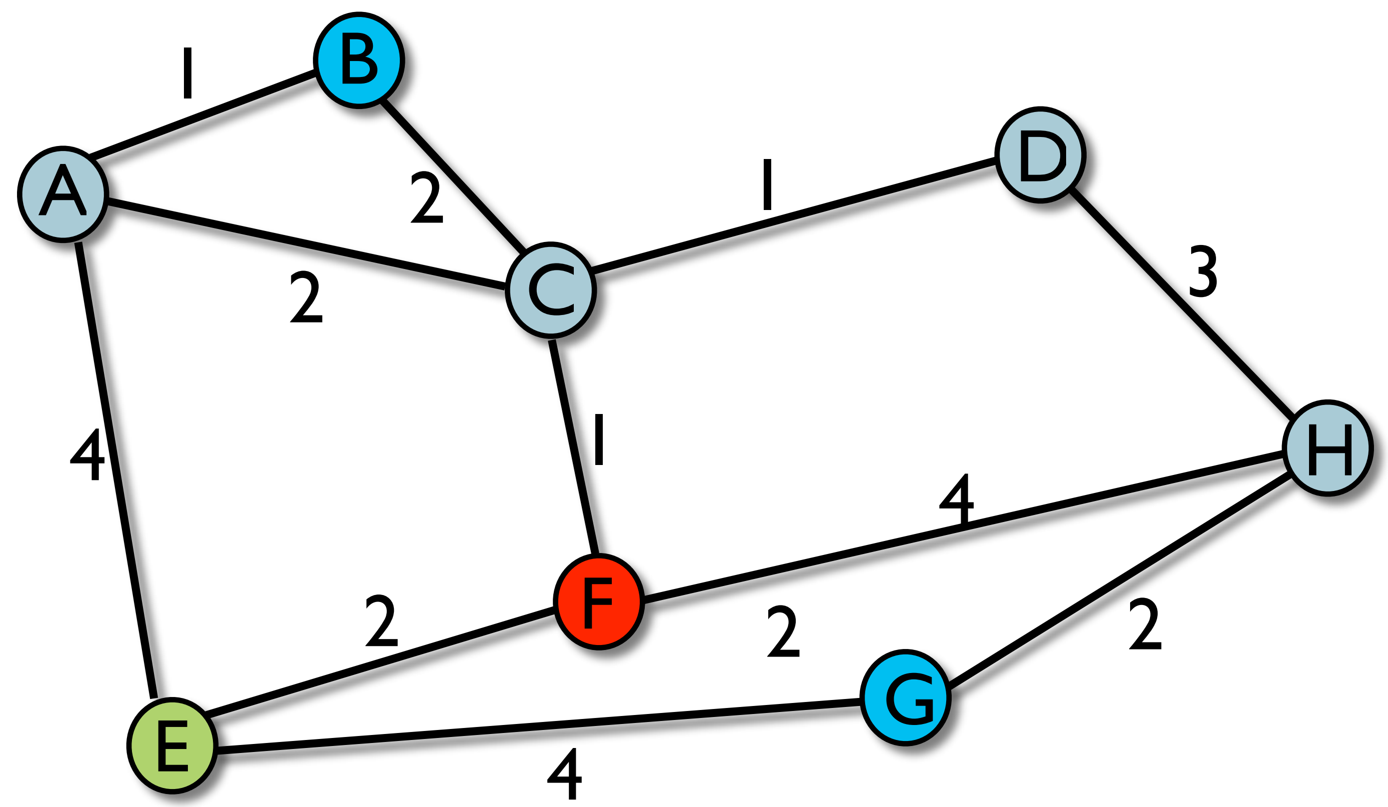
while $(w \notin B(v))$

$i \leftarrow i+1$

$w \leftarrow p_i(v)$

$(u,v) \leftarrow (v,u)$

return $\delta(w,u) + \delta(w,v)$



- **prove** $\text{dist}_k(u,v) \leq (2k-1) \cdot \delta(v,u)$
  - every iteration increases $\delta(w,u)$ by $\leq \delta(v,u) = \Delta$
  - $\Rightarrow \delta(w,u) \leq (k-1) \cdot \Delta \Rightarrow \text{dist}_k(u,v) = \delta(w,u) + \delta(w,v) \Rightarrow$ ✓