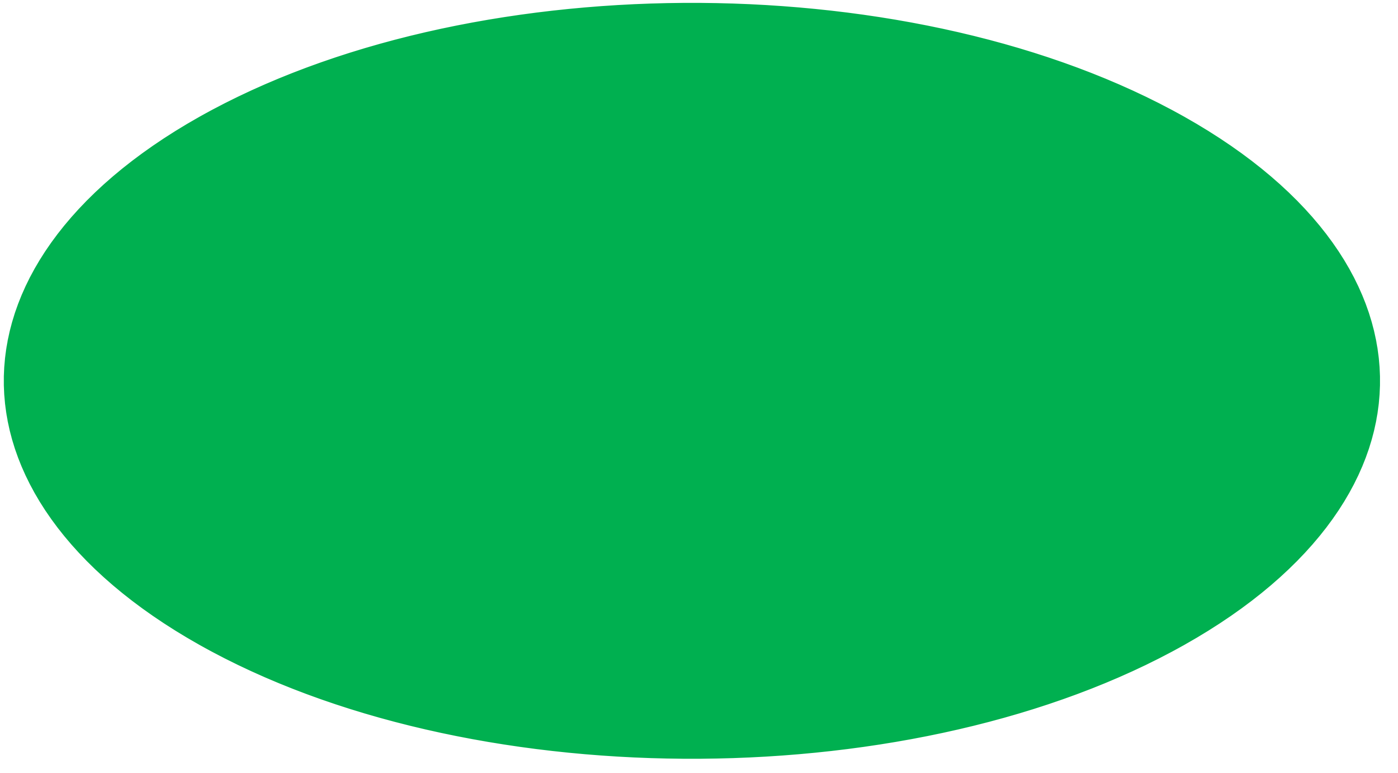
Fault-Tolerant Distance Labeling for Planar Graphs

Aviv Bar-Natan,
Panagiotis Charalampopoulos,
Paweł Gawrychowski,
Shay Mozes,
Oren Weimann

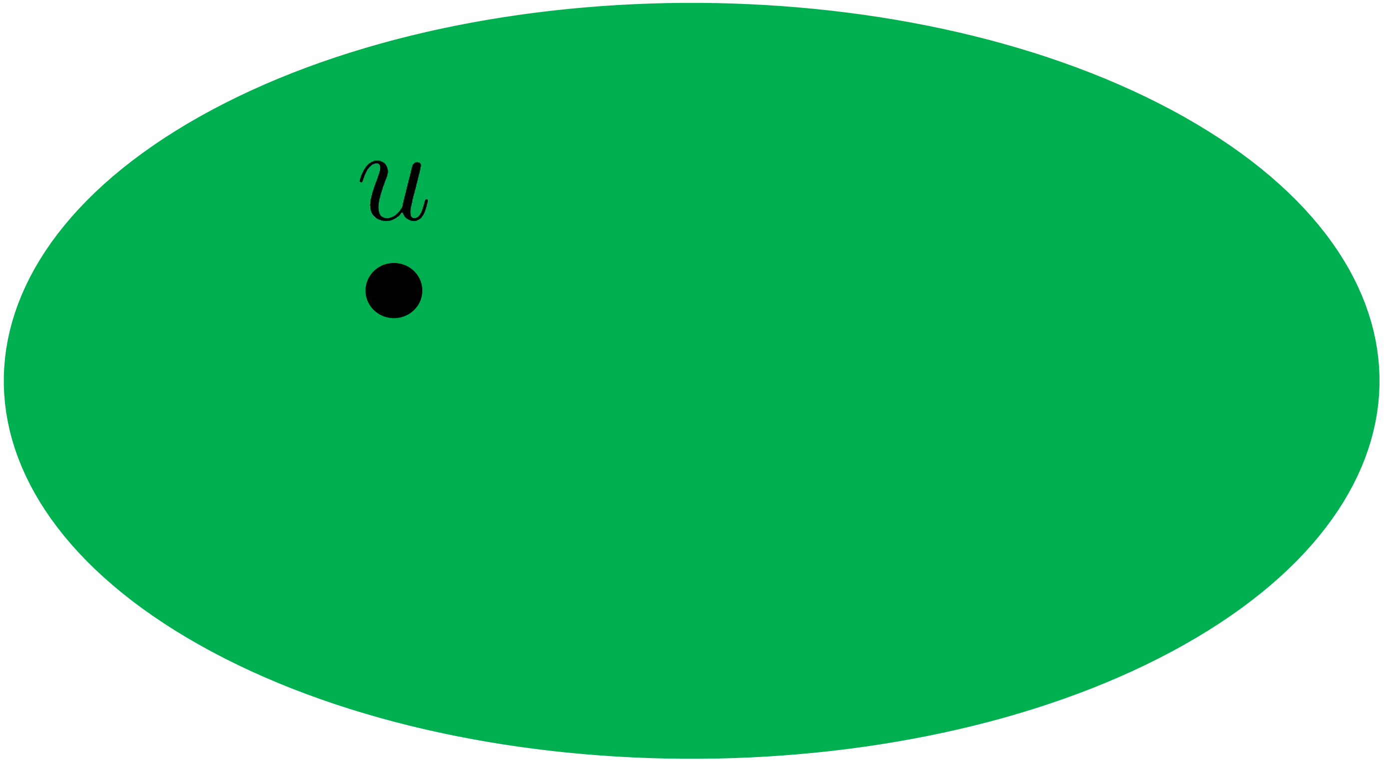
Slides by Aviv Bar-Natan

Distance Labeling

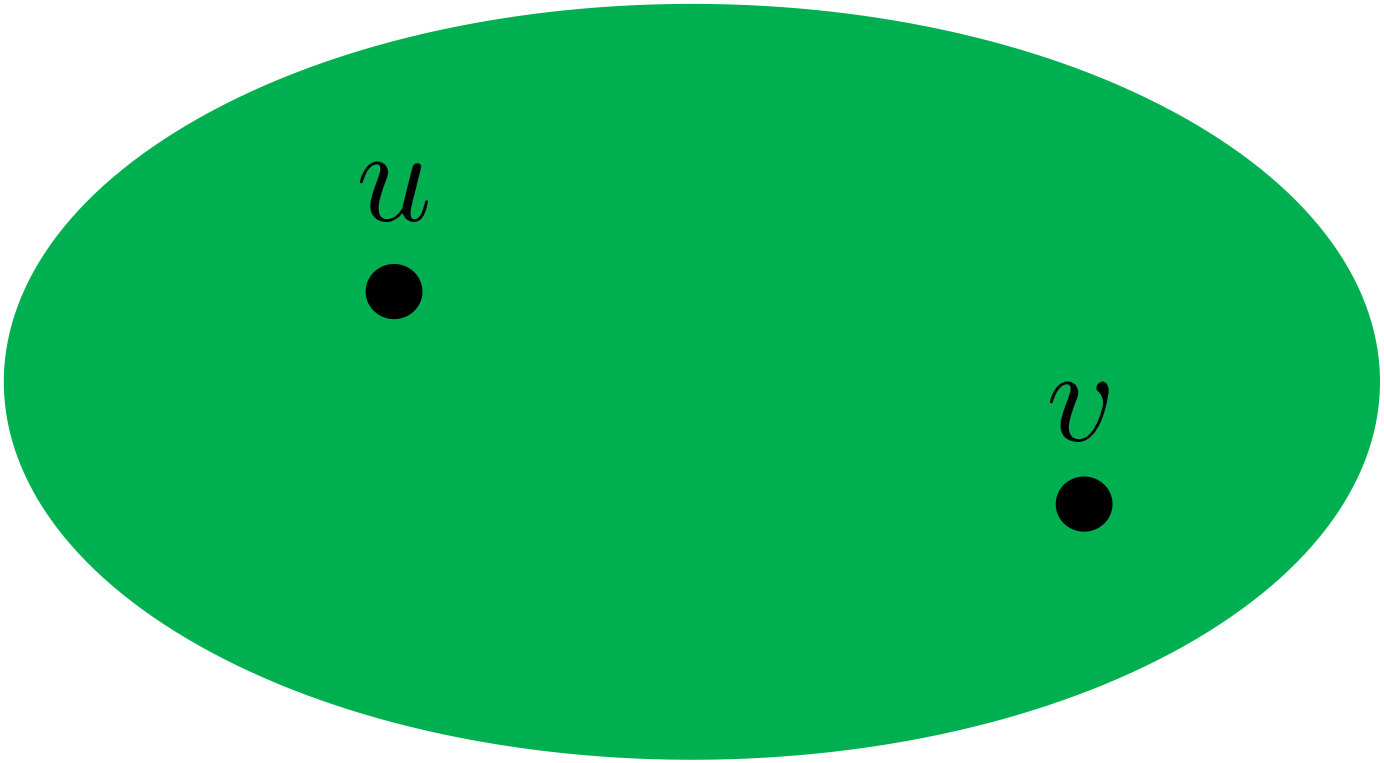
Distance Labeling



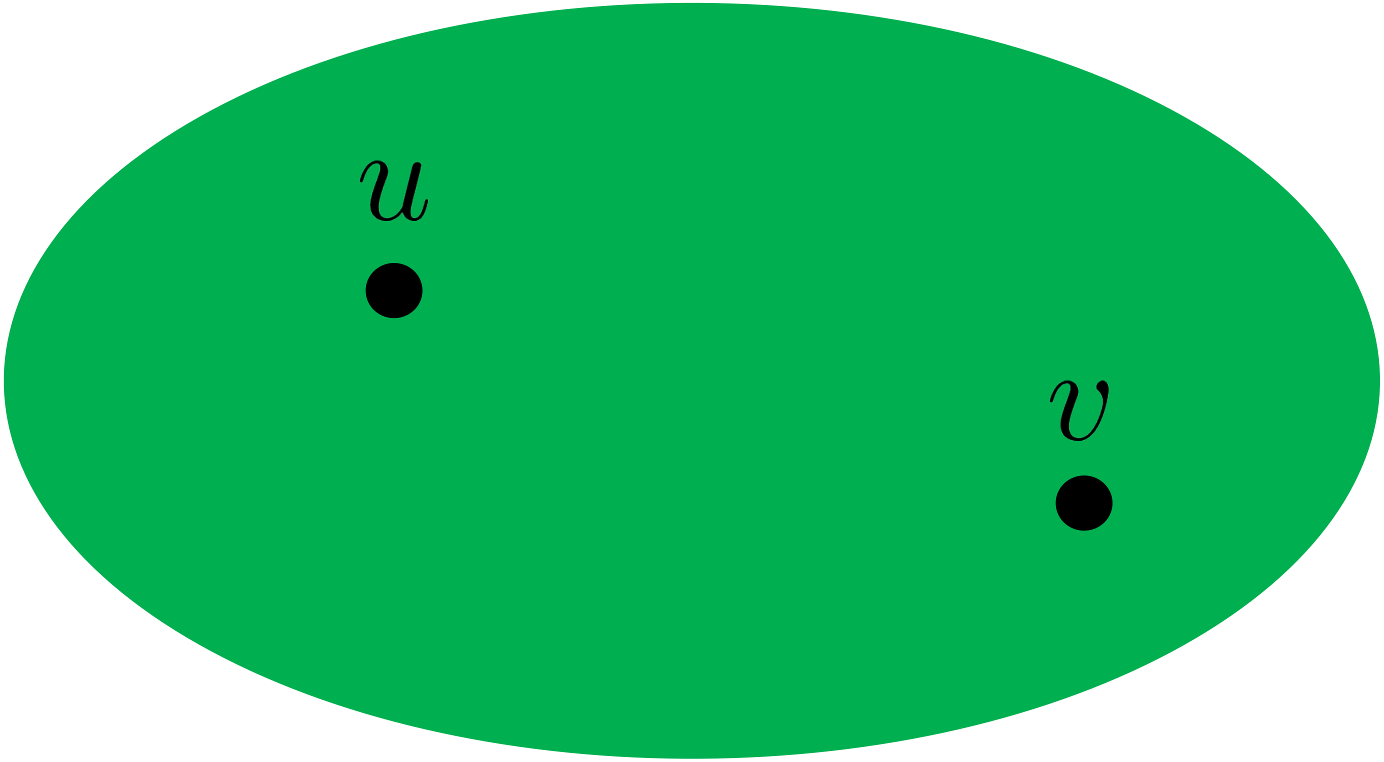
Distance Labeling



Distance Labeling



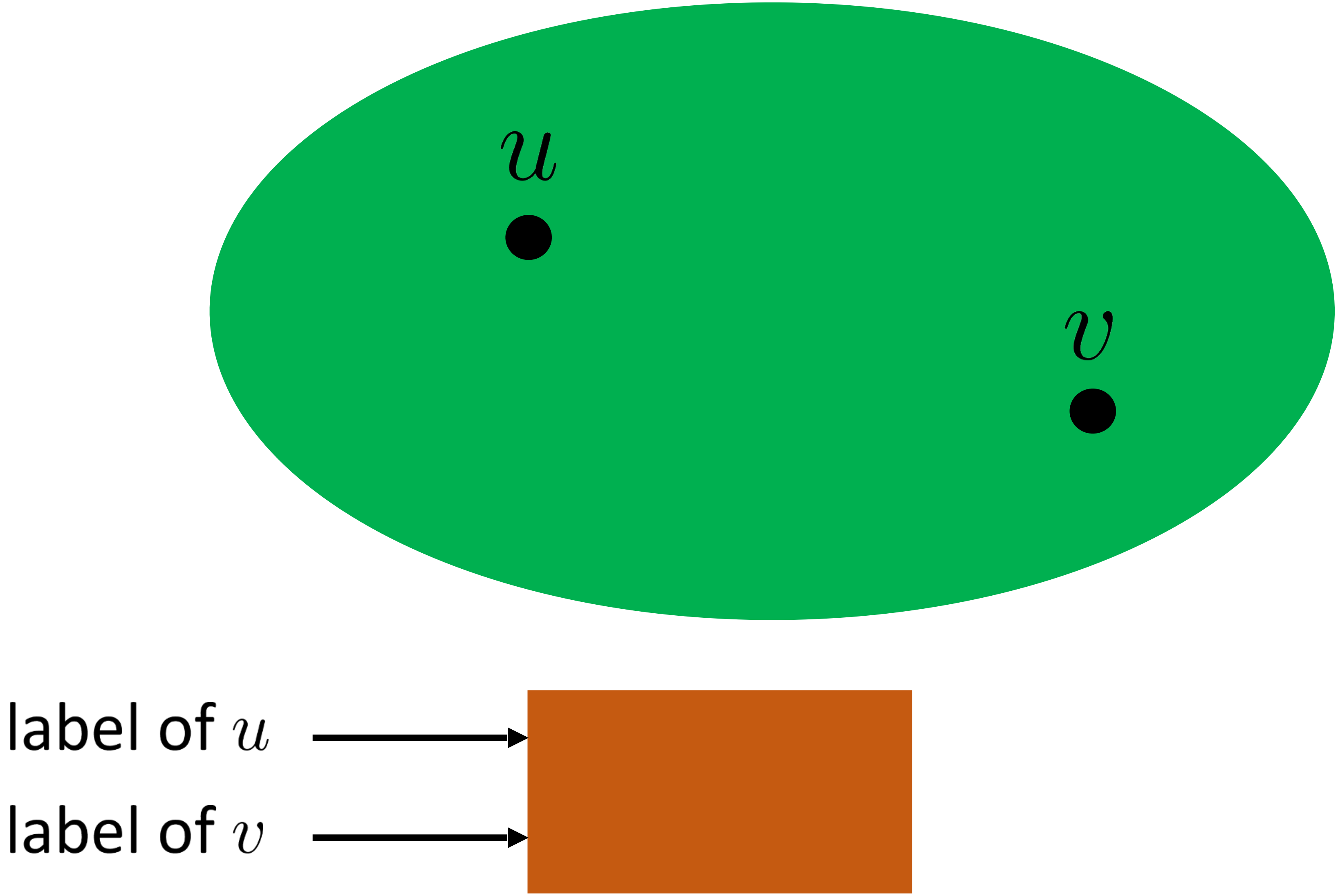
Distance Labeling



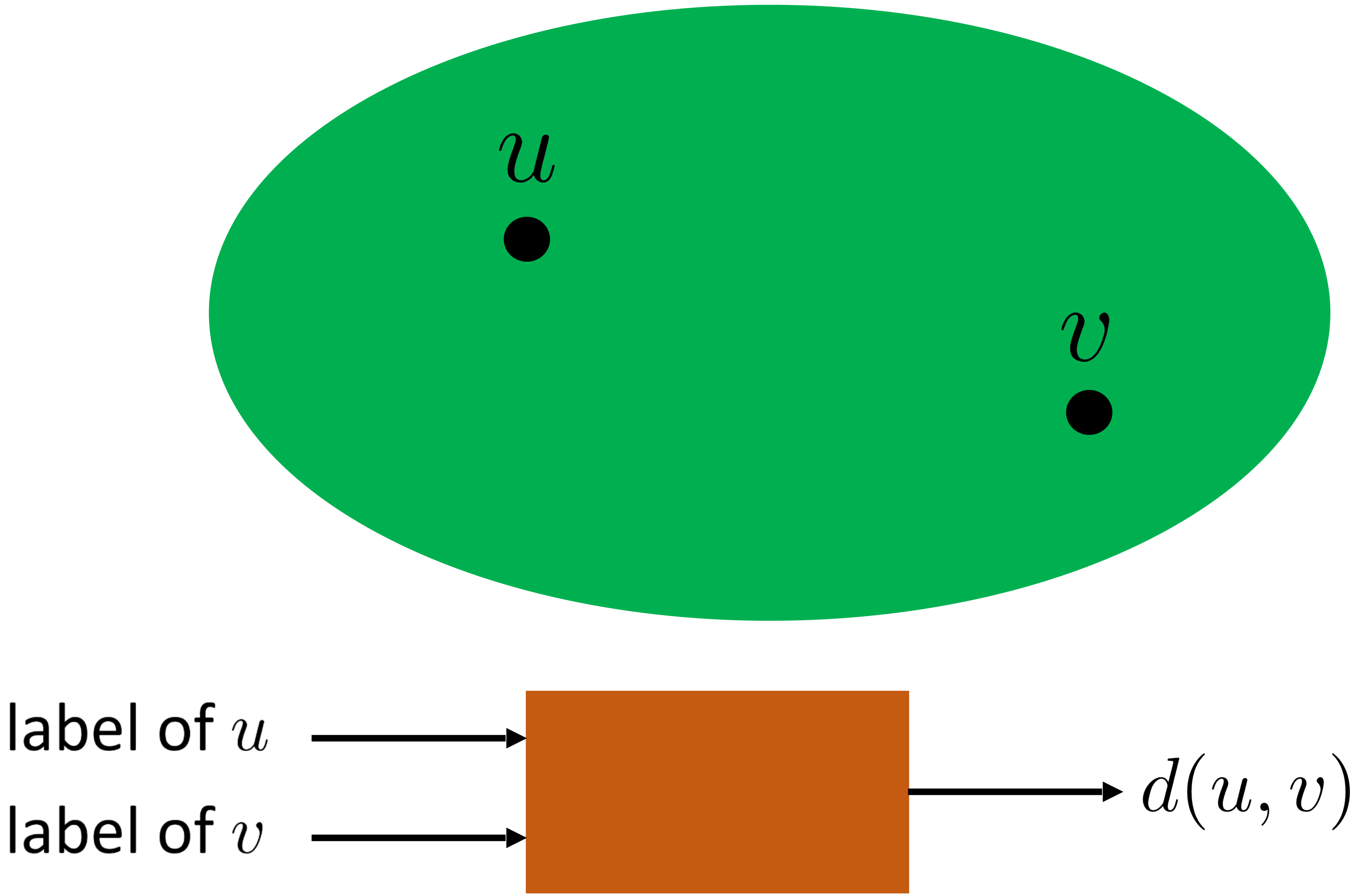
label of u

label of v

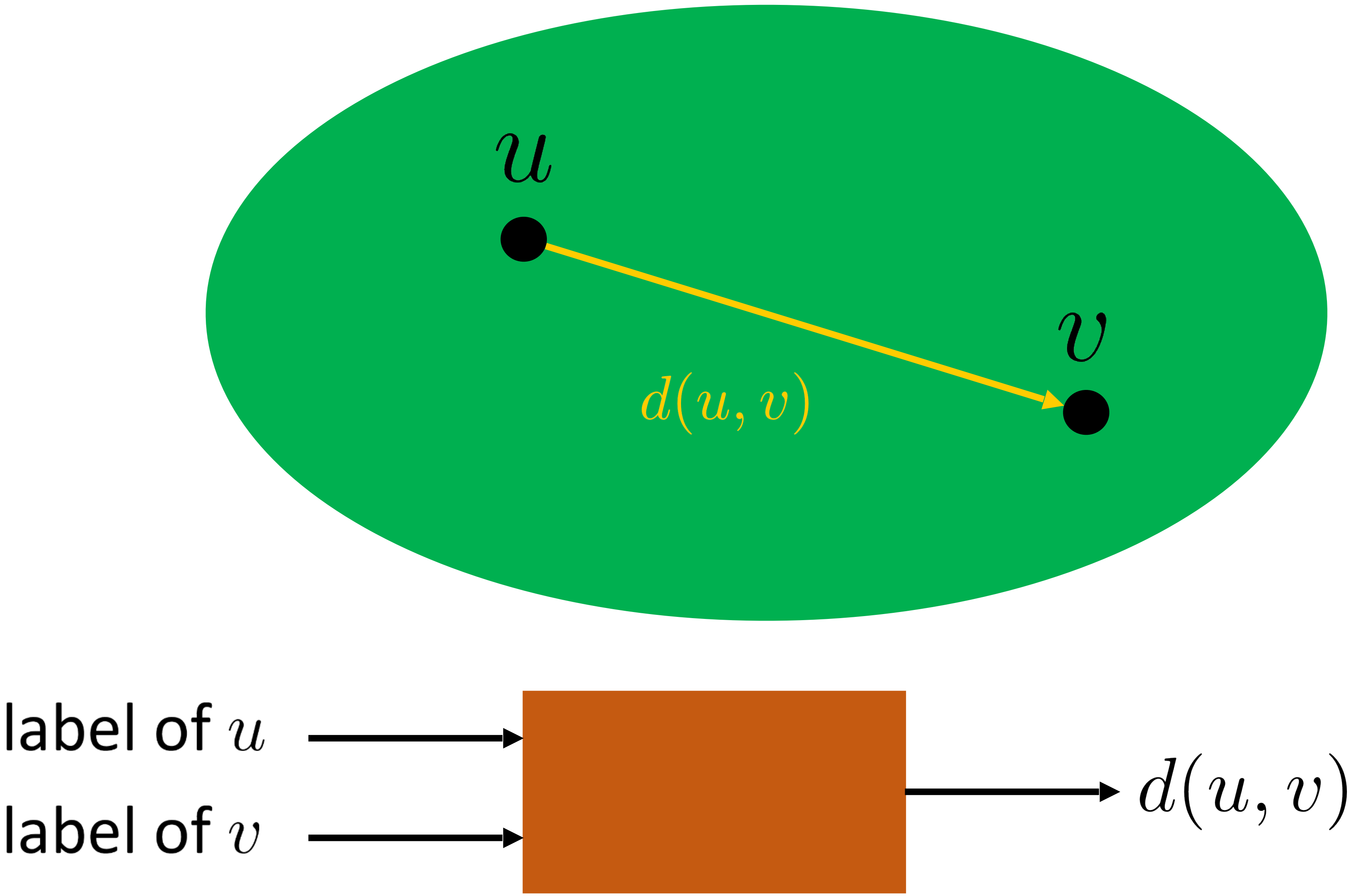
Distance Labeling



Distance Labeling



Distance Labeling



Some Results

Some Results

- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

Some Results

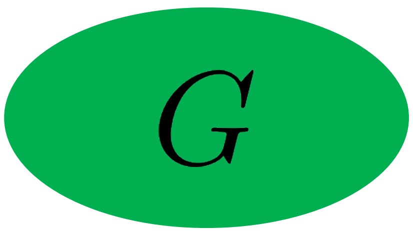
- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

Some Results

- General graphs: $\tilde{\Theta}(n)$

[Gavoille et al. SODA '01]



$$L(v_1), ..., L(v_n)$$

Some Results

- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

$$L(v_1), \ldots, L(v_n)$$

$2^{\binom{n}{2}}$ different strings

Some Results

- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

$L(v_1), \ldots, L(v_n)$

$2^{\binom{n}{2}}$ different strings

string size is $\log(2^{\binom{n}{2}}) = \Omega(n^2)$

Some Results

- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

Some Results

- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

- Planar graphs: $O(\sqrt{n} \cdot \log n)$

  [Gavoille et al. SODA '01]

Some Results

- General graphs: $\tilde{\Theta}(n)$

  [Gavoille et al. SODA '01]

- Planar graphs: $O(\sqrt{n} \cdot \log n)$ Weighted: $\Omega(\sqrt{n})$

  [Gavoille et al. SODA '01]

Some Results

- General graphs: $\tilde{\Theta}(n)$
  [Gavoille et al. SODA '01]

- Planar graphs: $O(\sqrt{n} \cdot \log n)$
  [Gavoille et al. SODA '01]

  Weighted: $\Omega(\sqrt{n})$

  Unweighted: $\Omega(n^{1/3})$

Some Results

- General graphs: $\tilde{\Theta}(n)$
  [Gavoille et al. SODA '01]

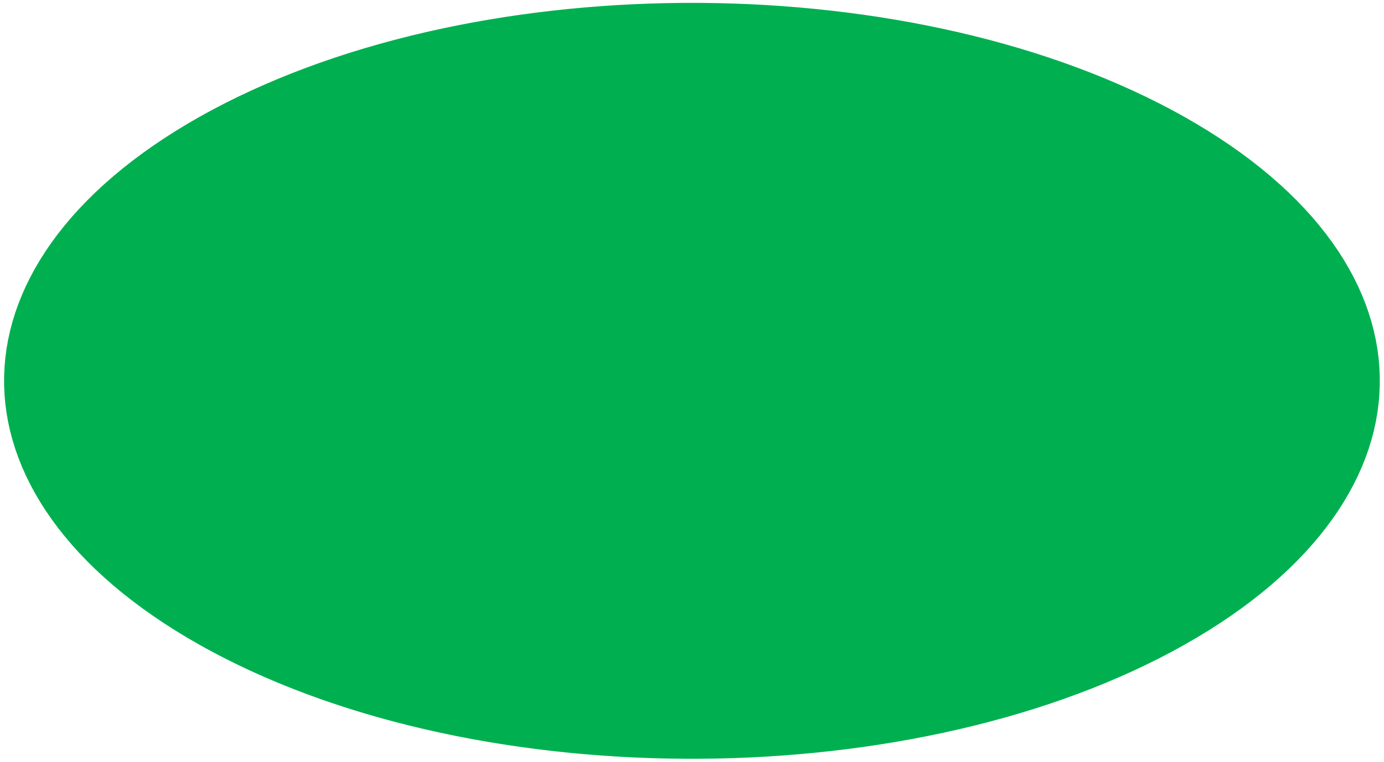
- Planar graphs: $O(\sqrt{n} \cdot \log n)$
  [Gavoille et al. SODA '01]

  Weighted: $\Omega(\sqrt{n})$

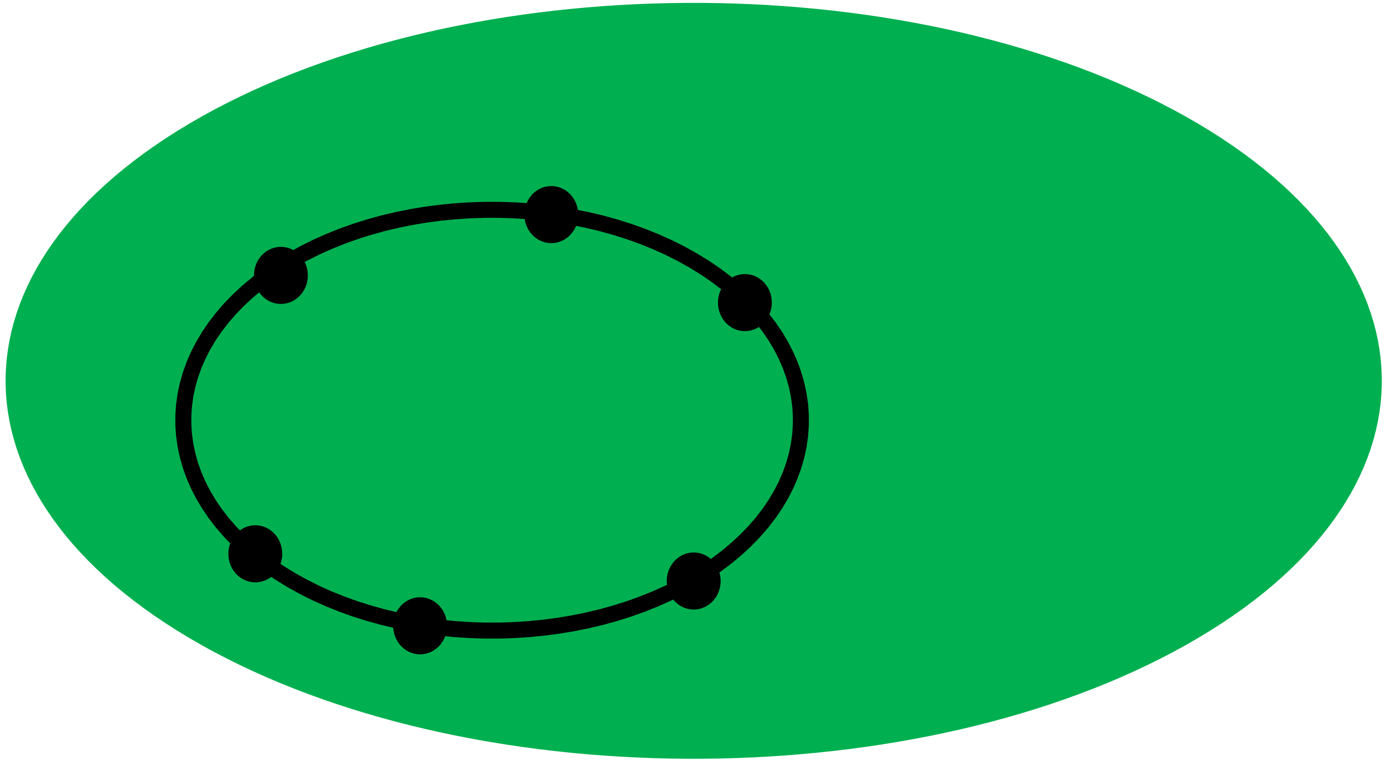
  Unweighted: $\Omega(n^{1/3})$

- Planar graphs $(1+\epsilon)$-approximation: $O(\log n/\epsilon)$
  [Thorup 2004]

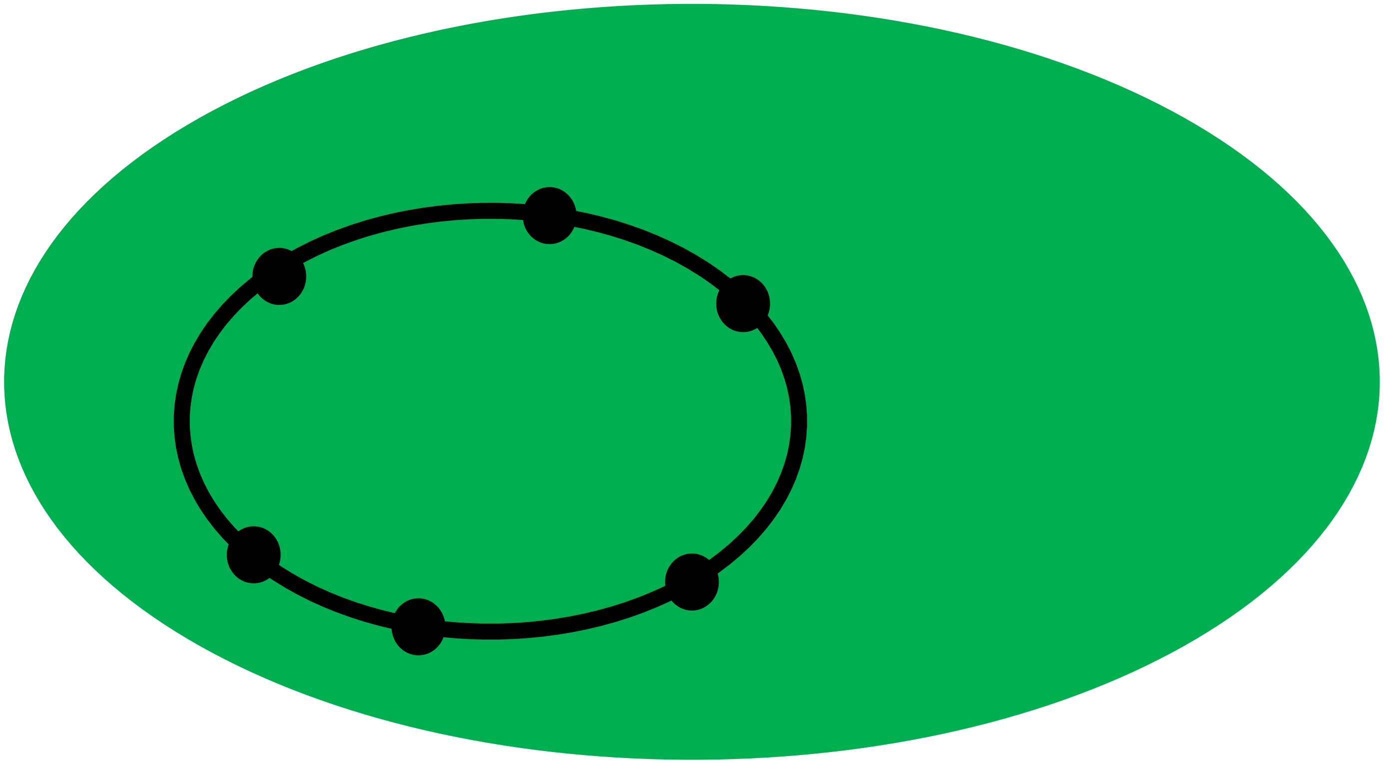
Small Separator



Small Separator

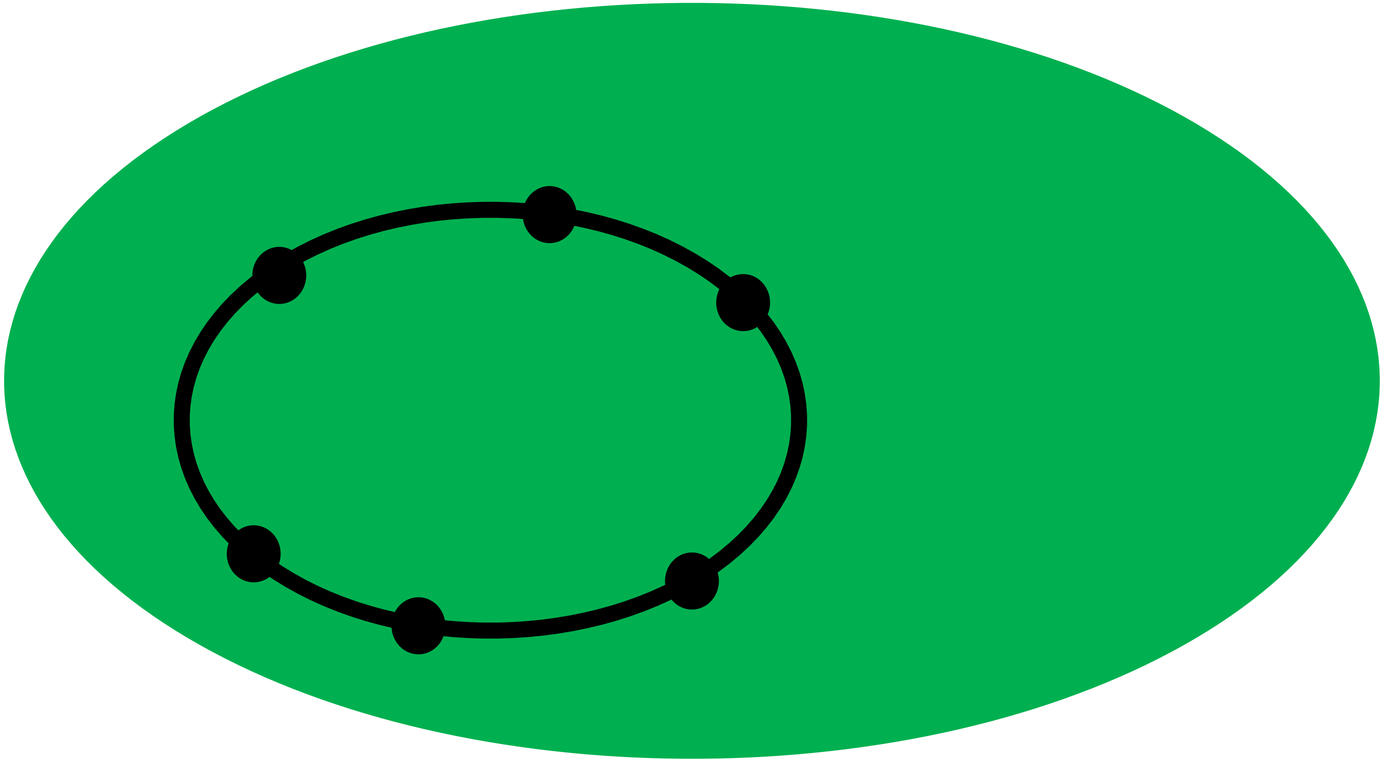


Small Separator



Seperator size $O(\sqrt{n})$

Small Separator

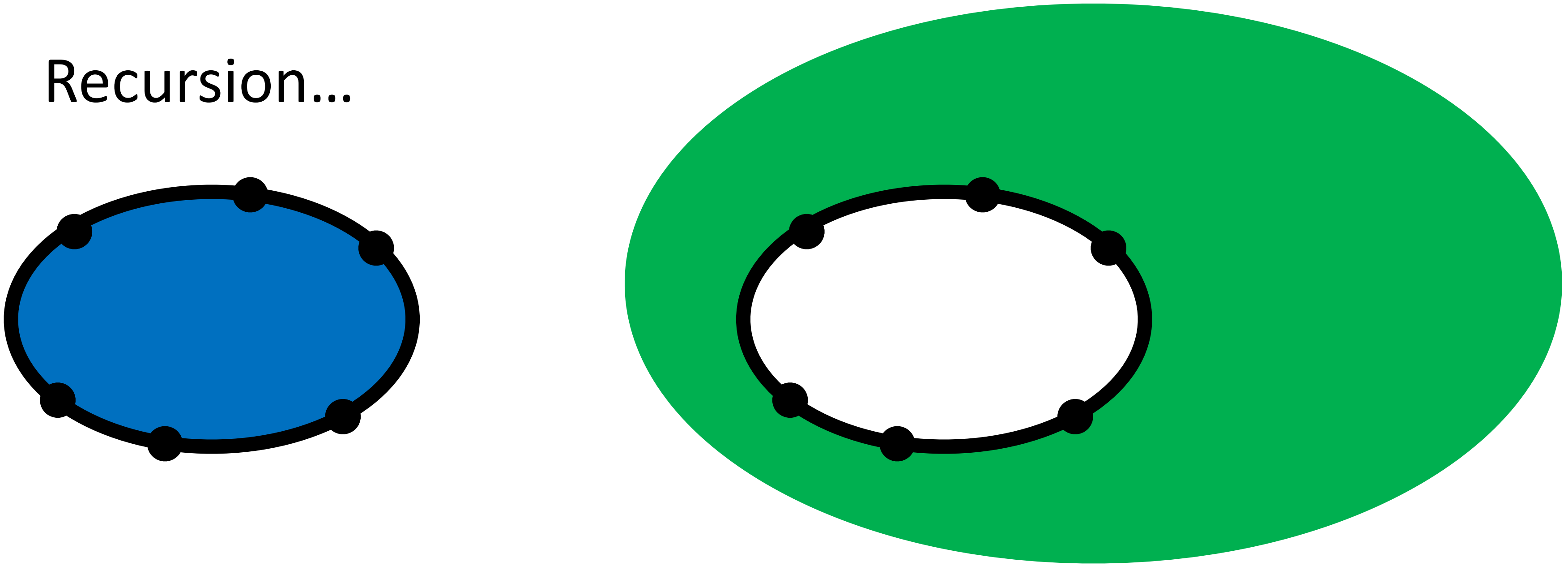


Seperator size $O(\sqrt{n})$

$$|green\ piece|, |blue\ piece| \leq 2n/3$$

Small Separator

Recursion…

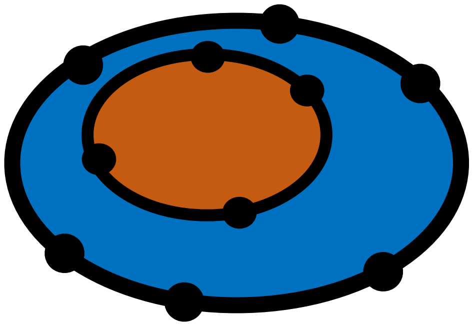


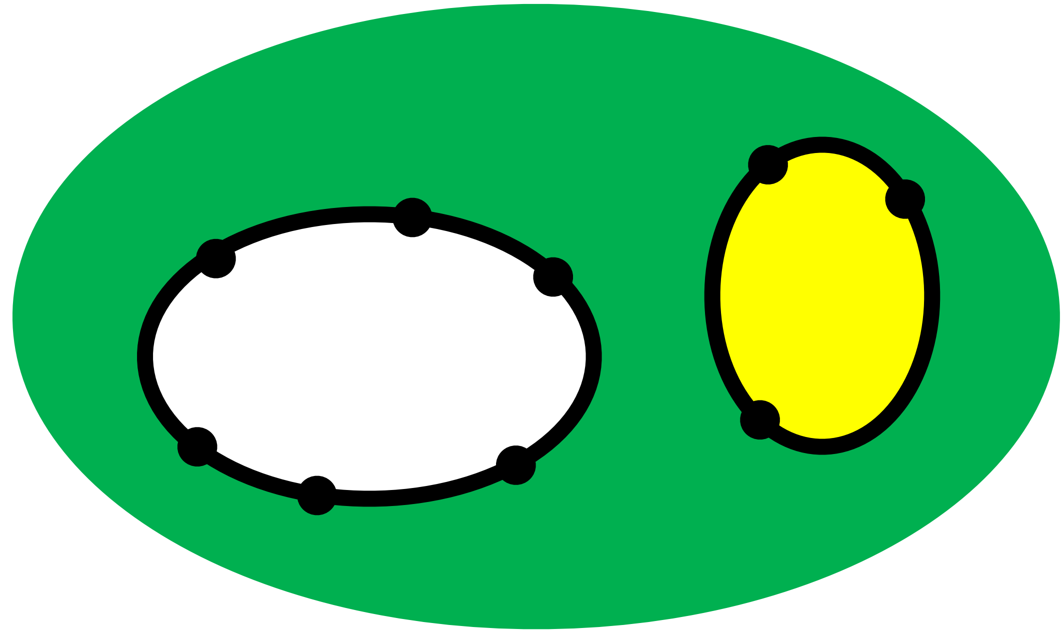
Seperator size $O(\sqrt{n})$

$$|green\ piece|, |blue\ piece| \leq 2n/3$$

Small Separator

Recursion…



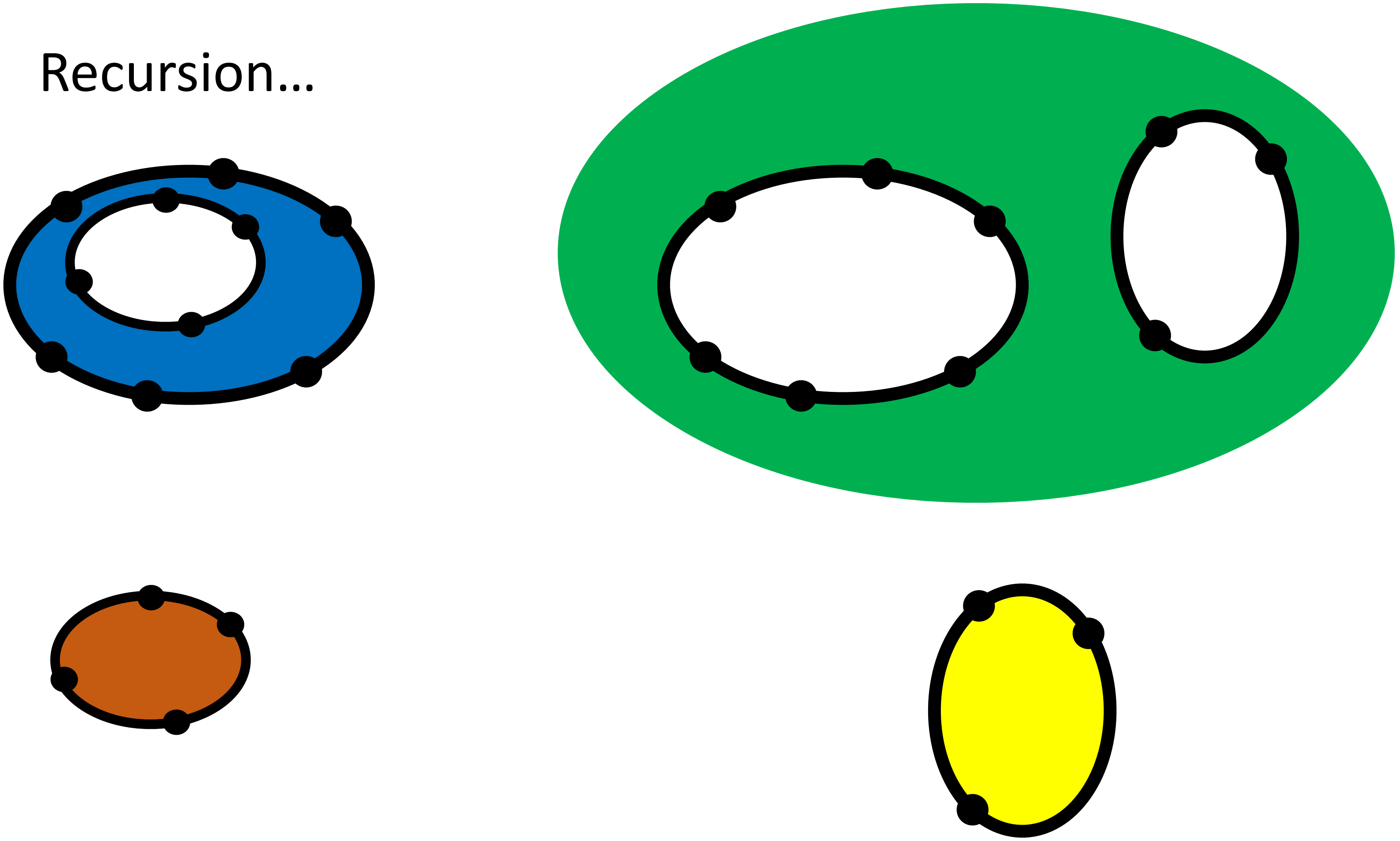


Seperator size $O(\sqrt{n})$

$$|green\ piece|, |blue\ piece| \leq 2n/3$$

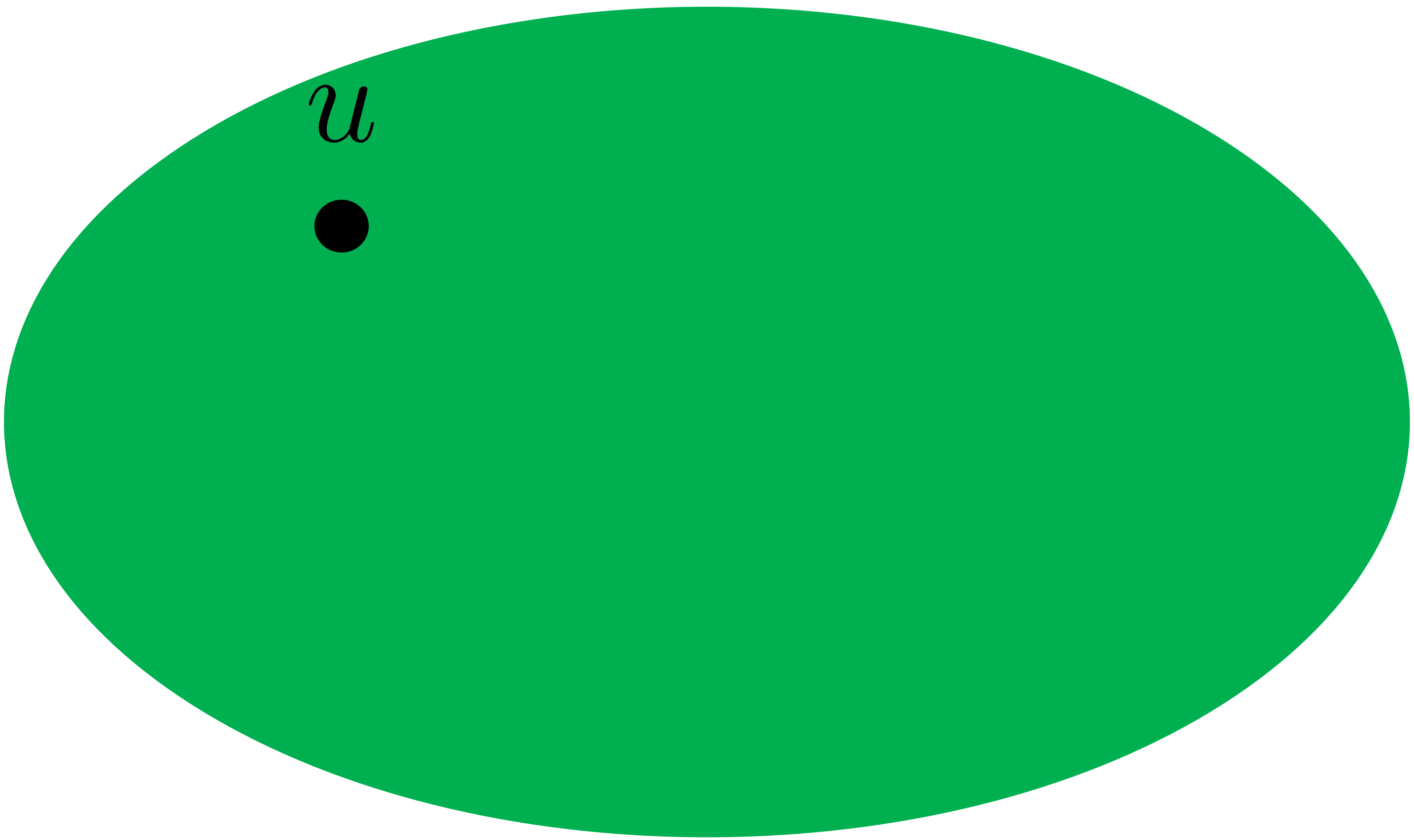
Small Separator

Recursion…



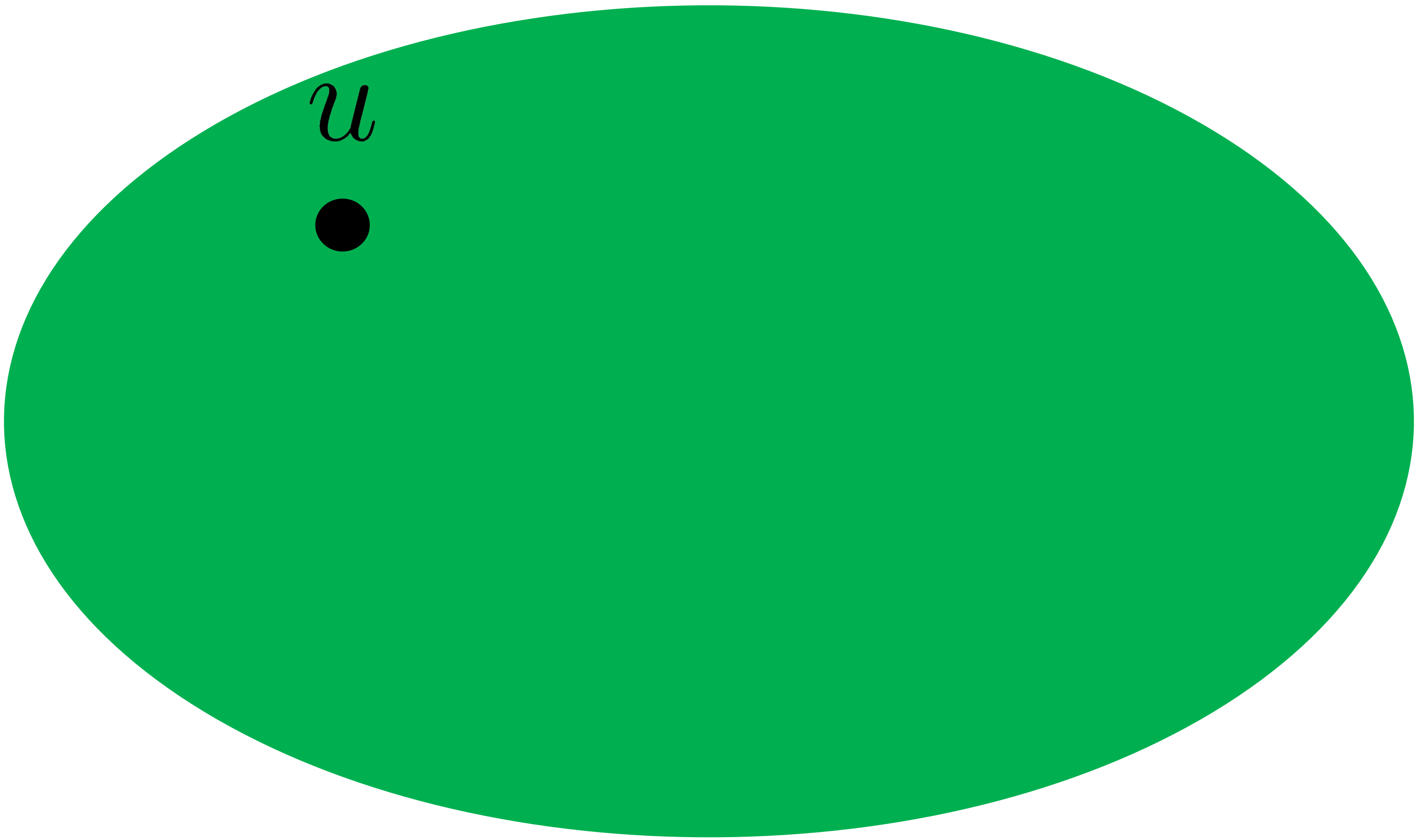
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



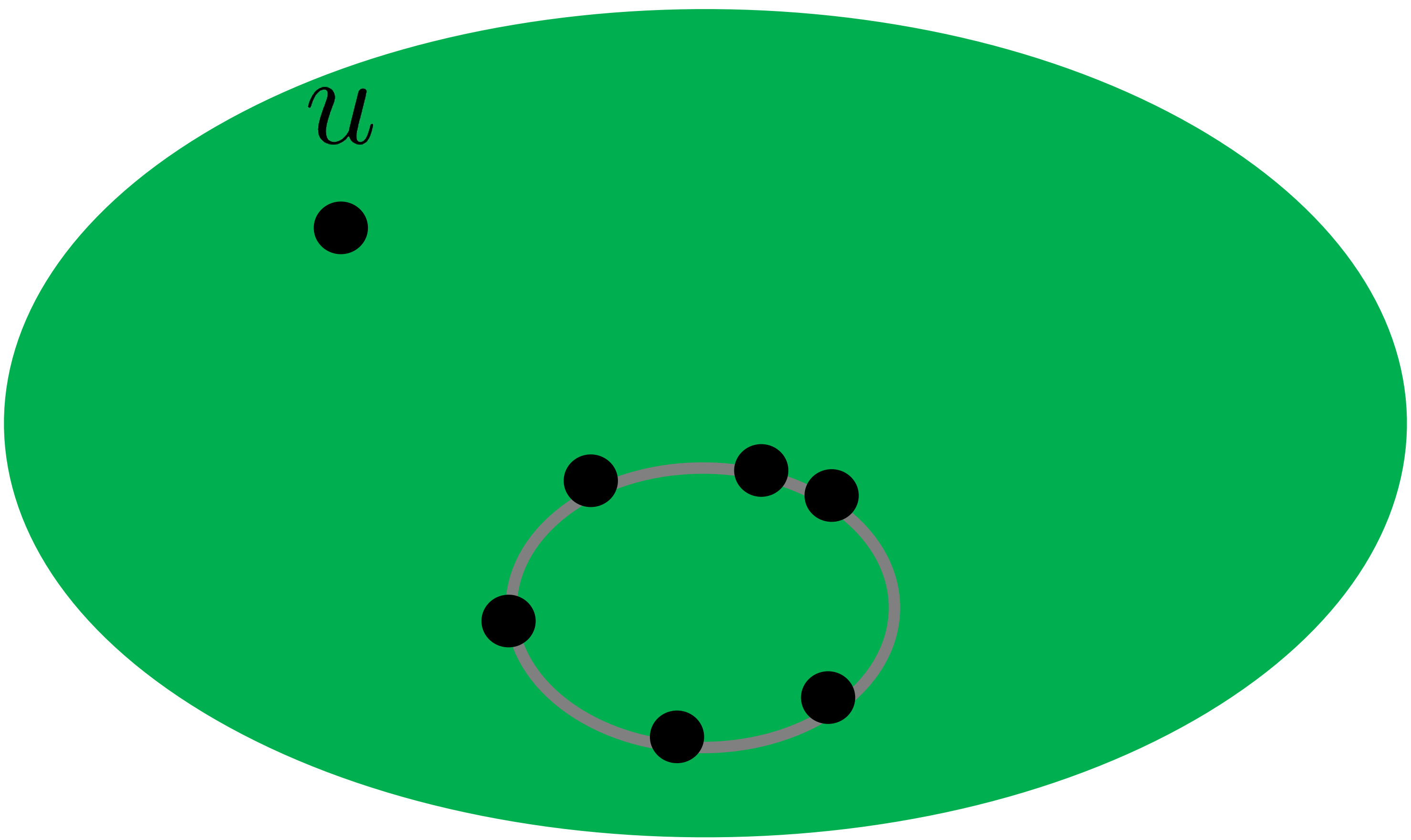
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



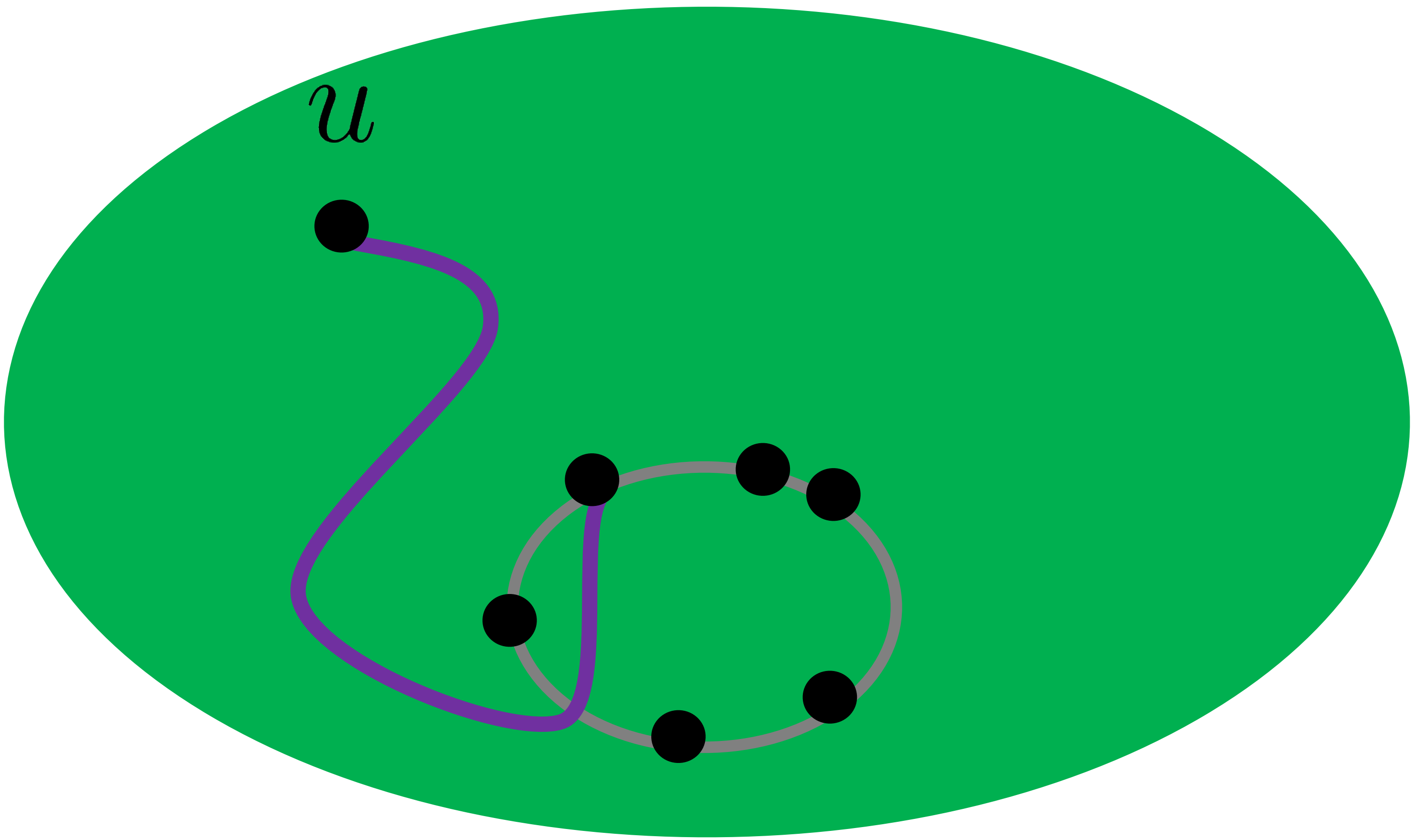
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



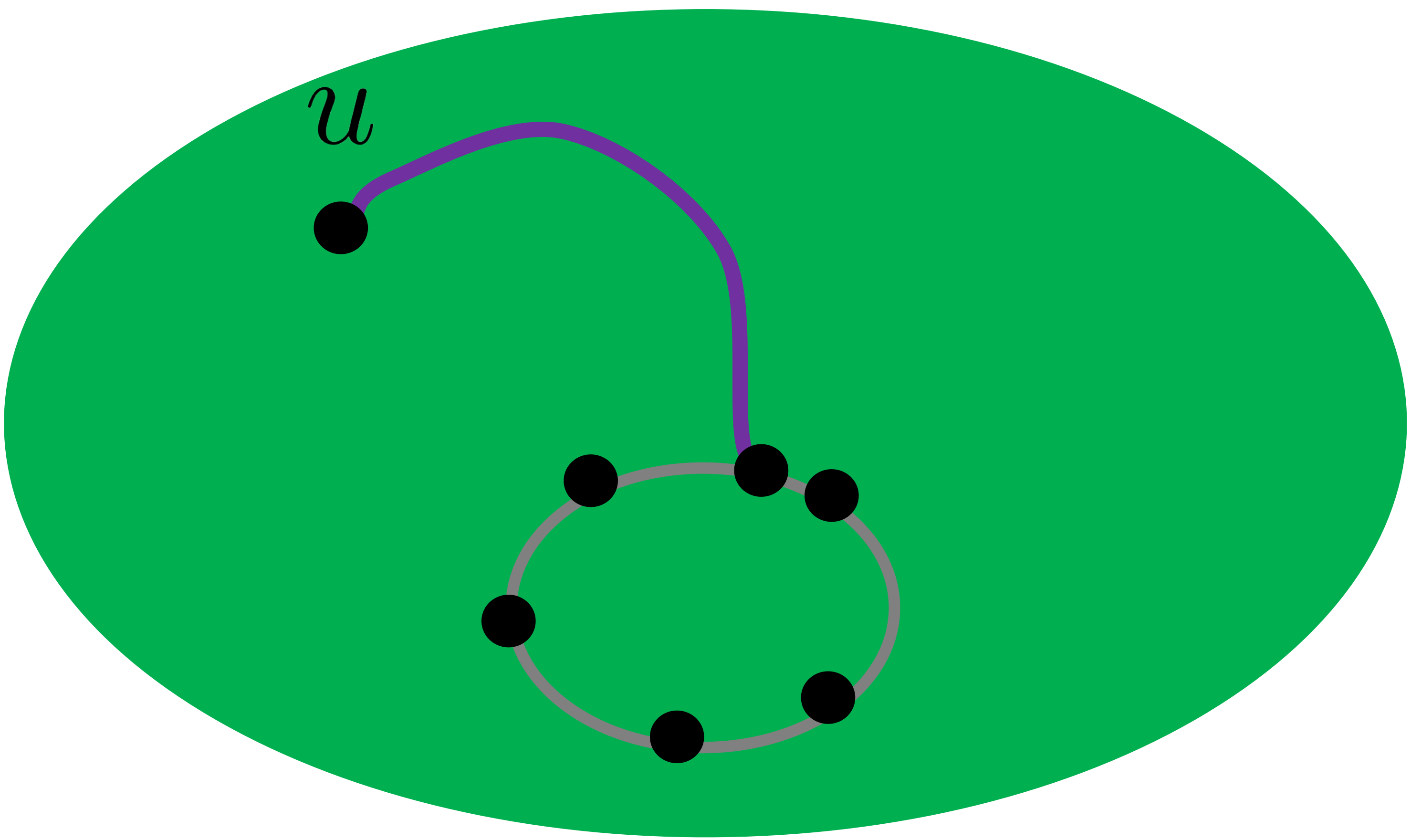
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



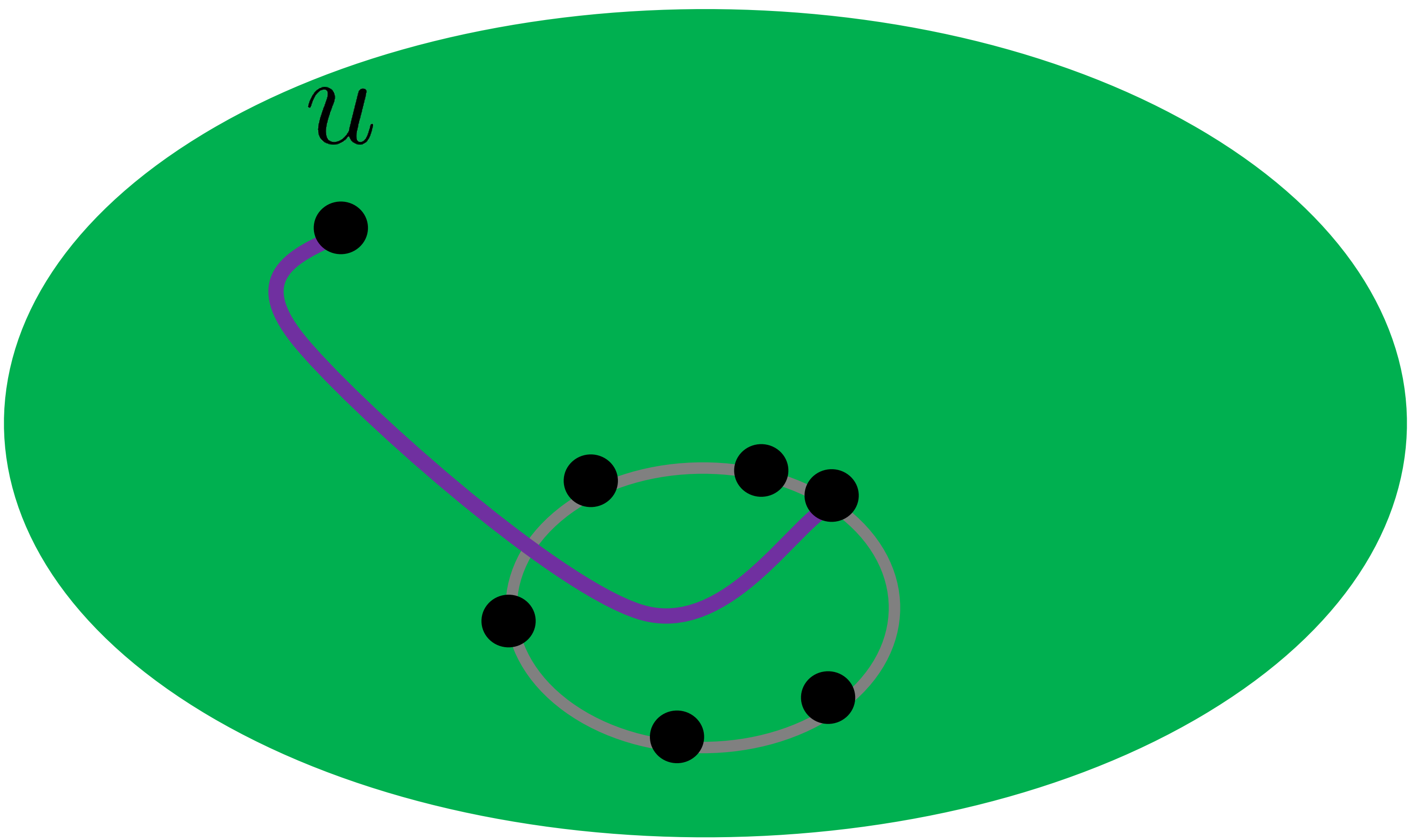
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



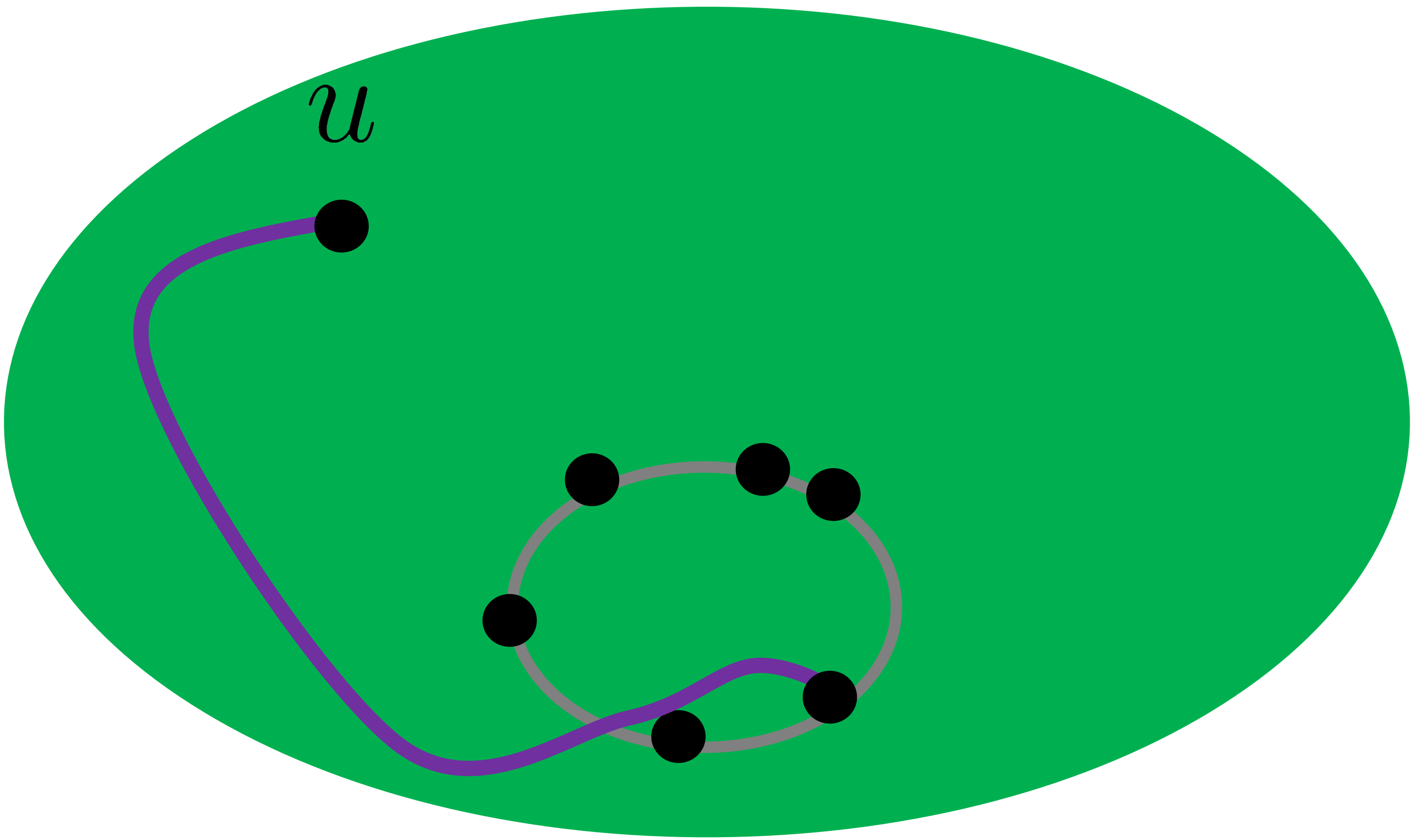
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



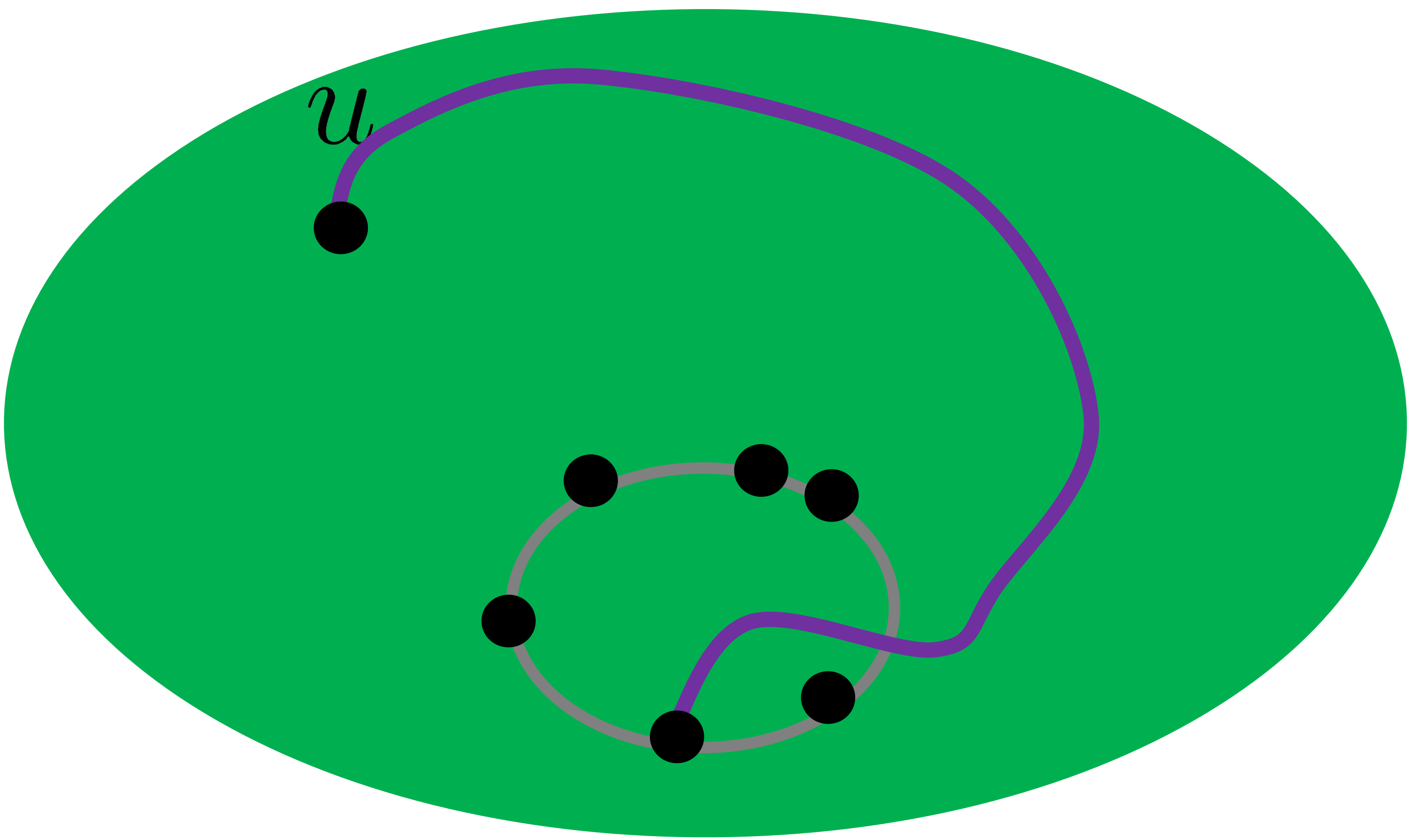
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



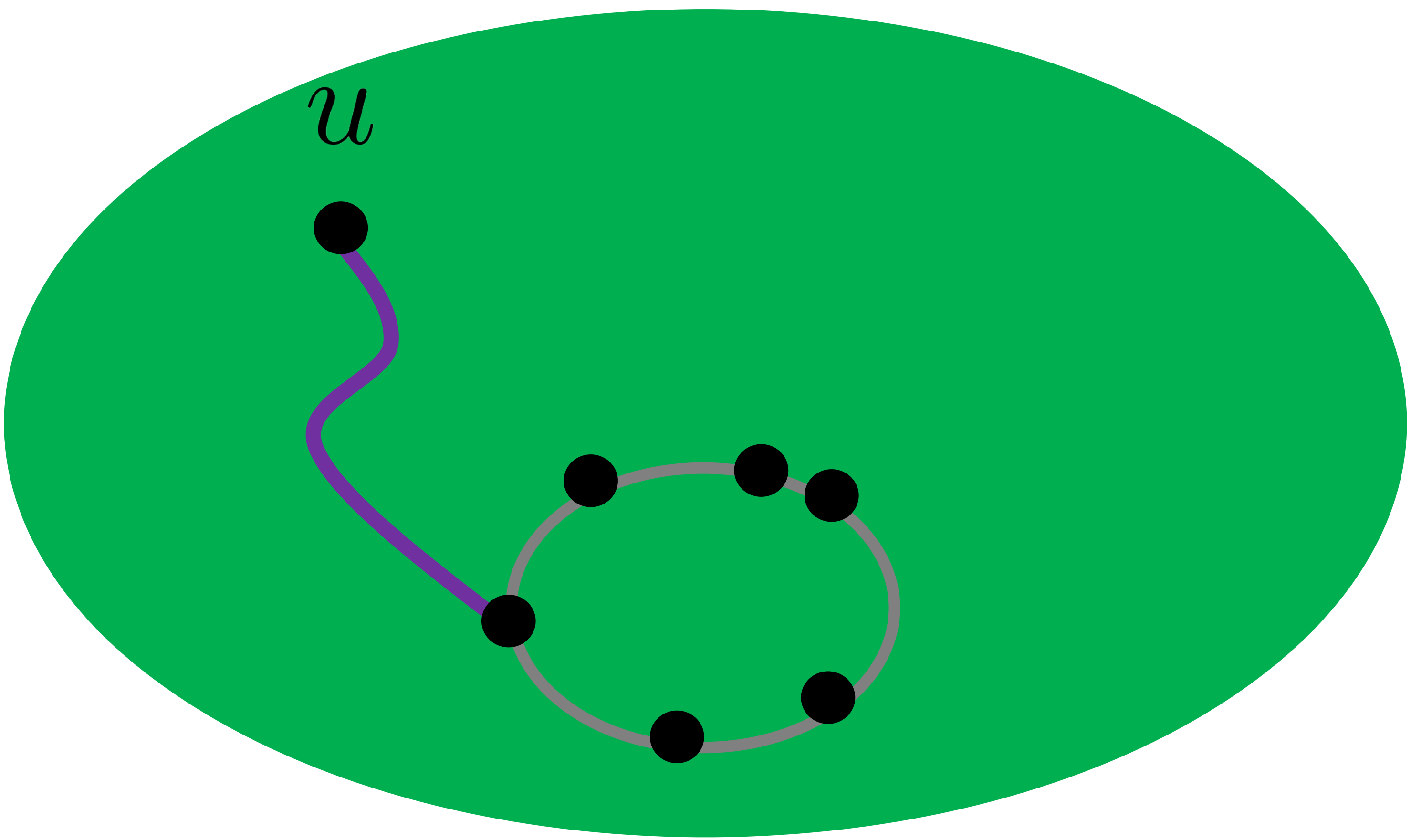
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



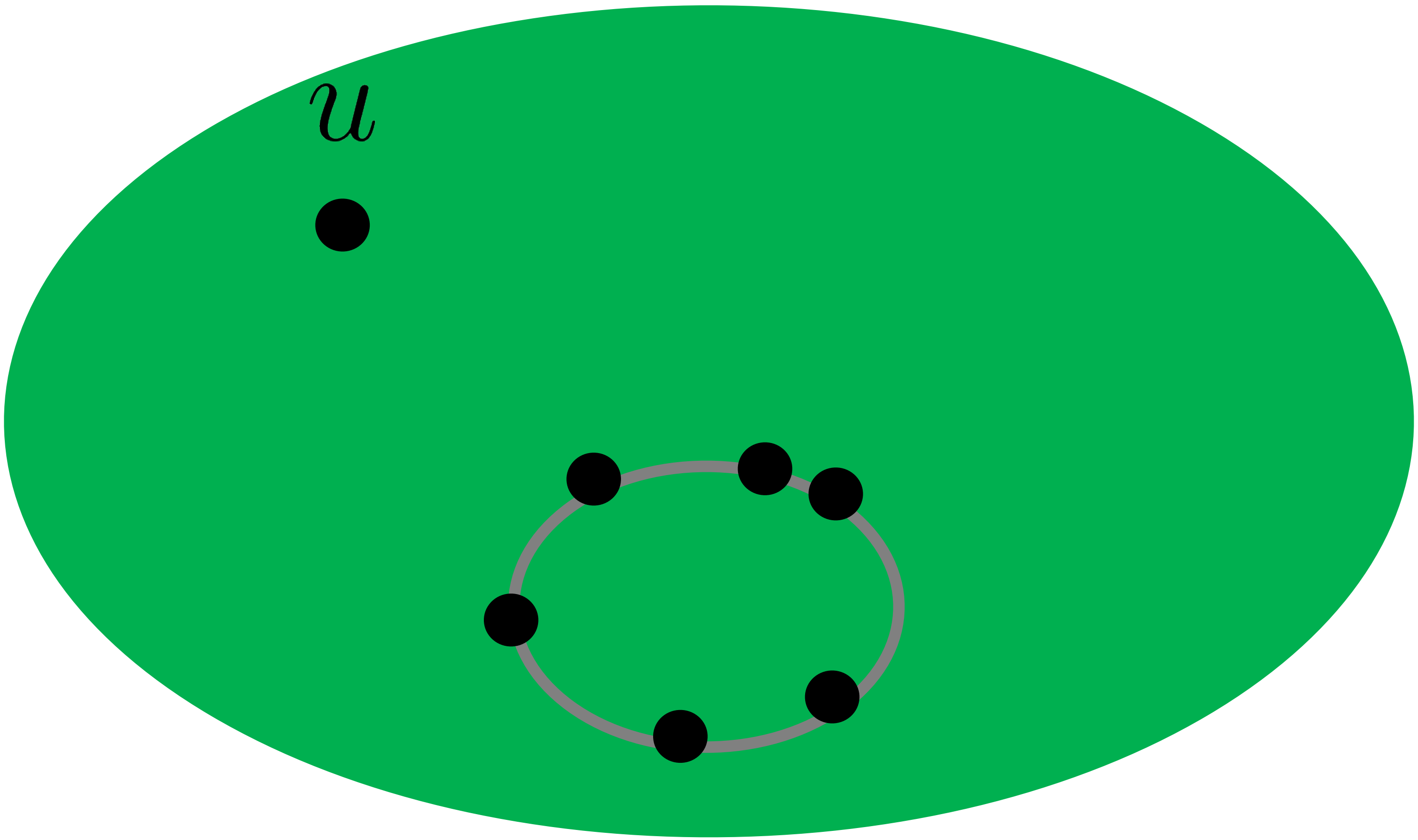
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



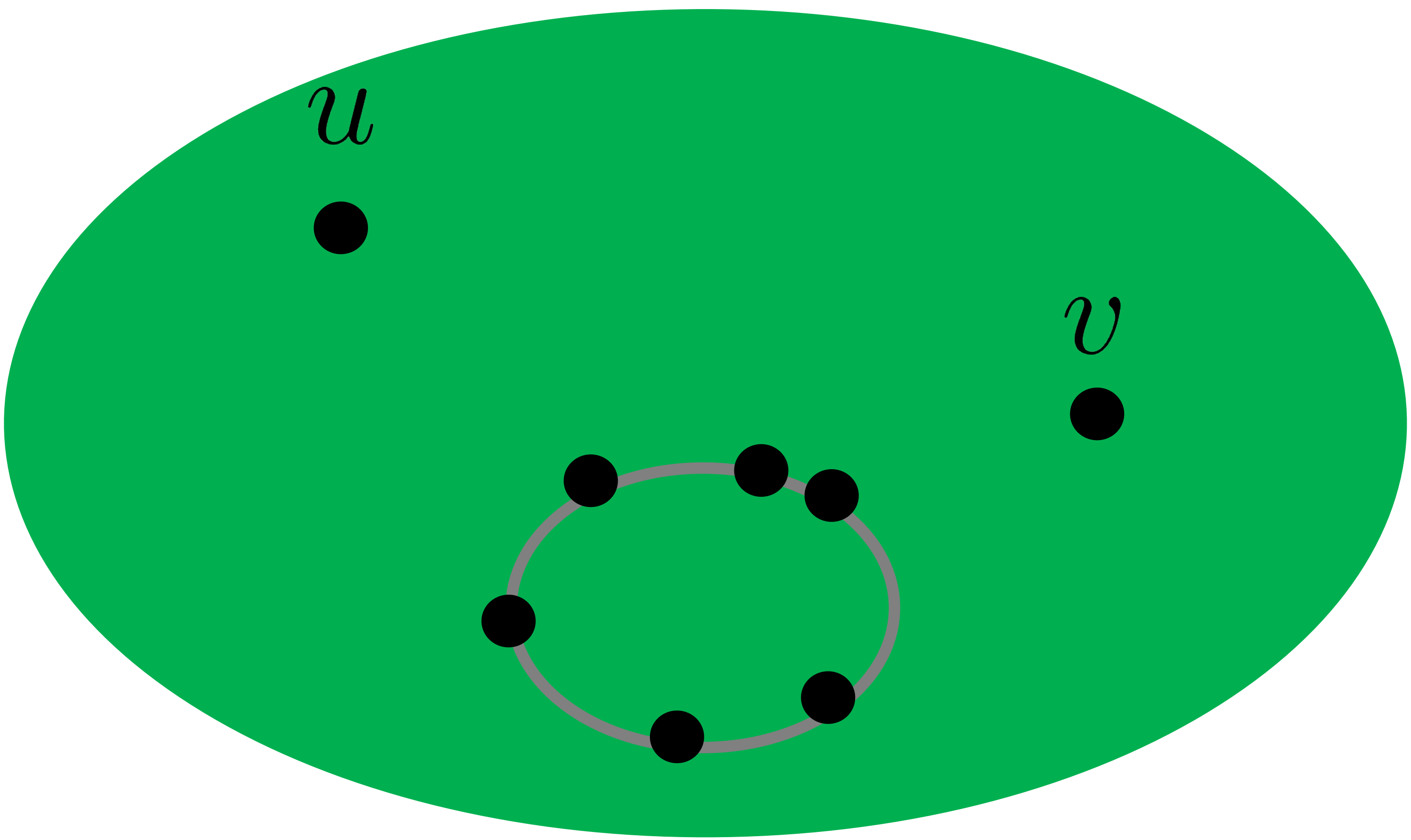
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u's label:



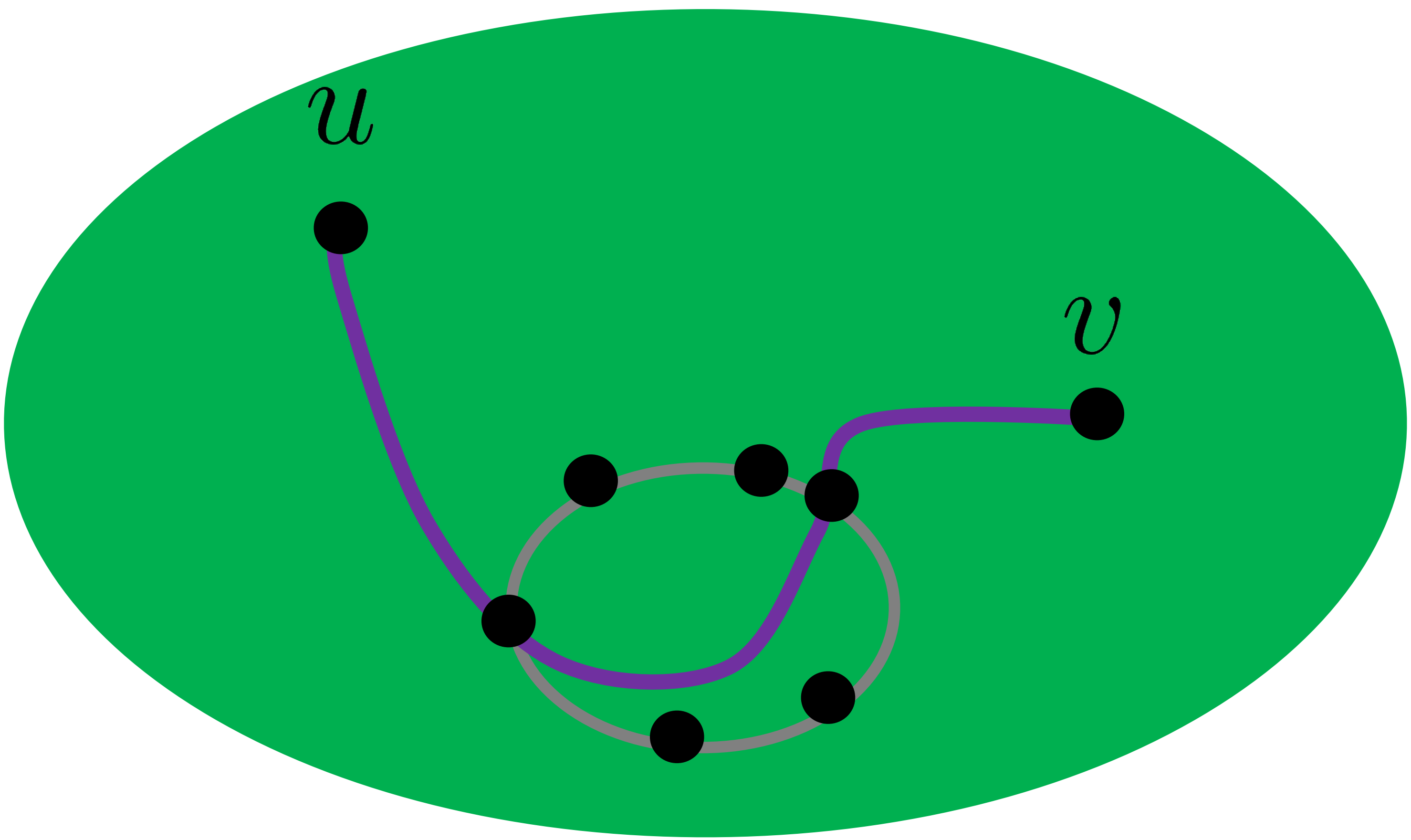
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u, v query:



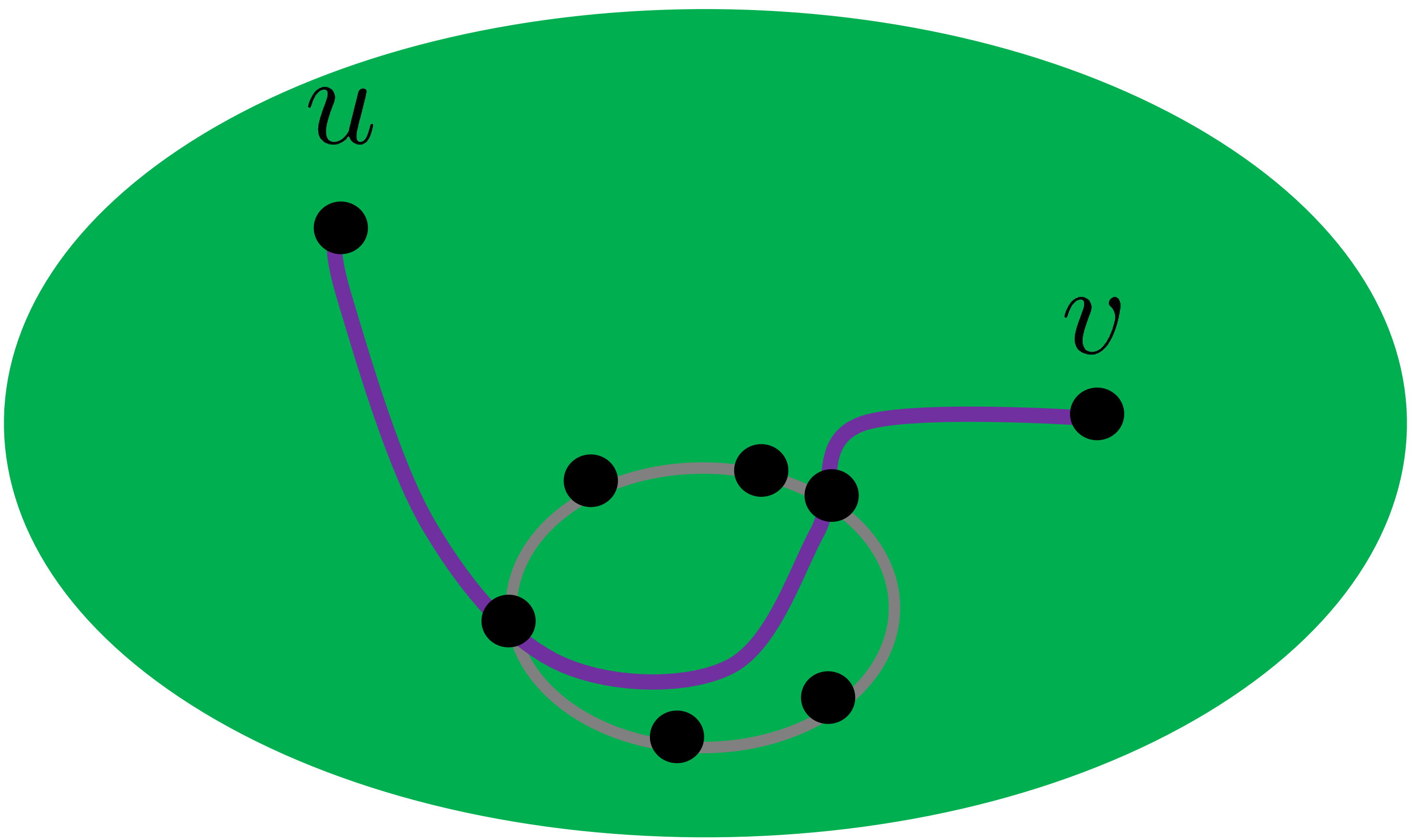
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u, v query:



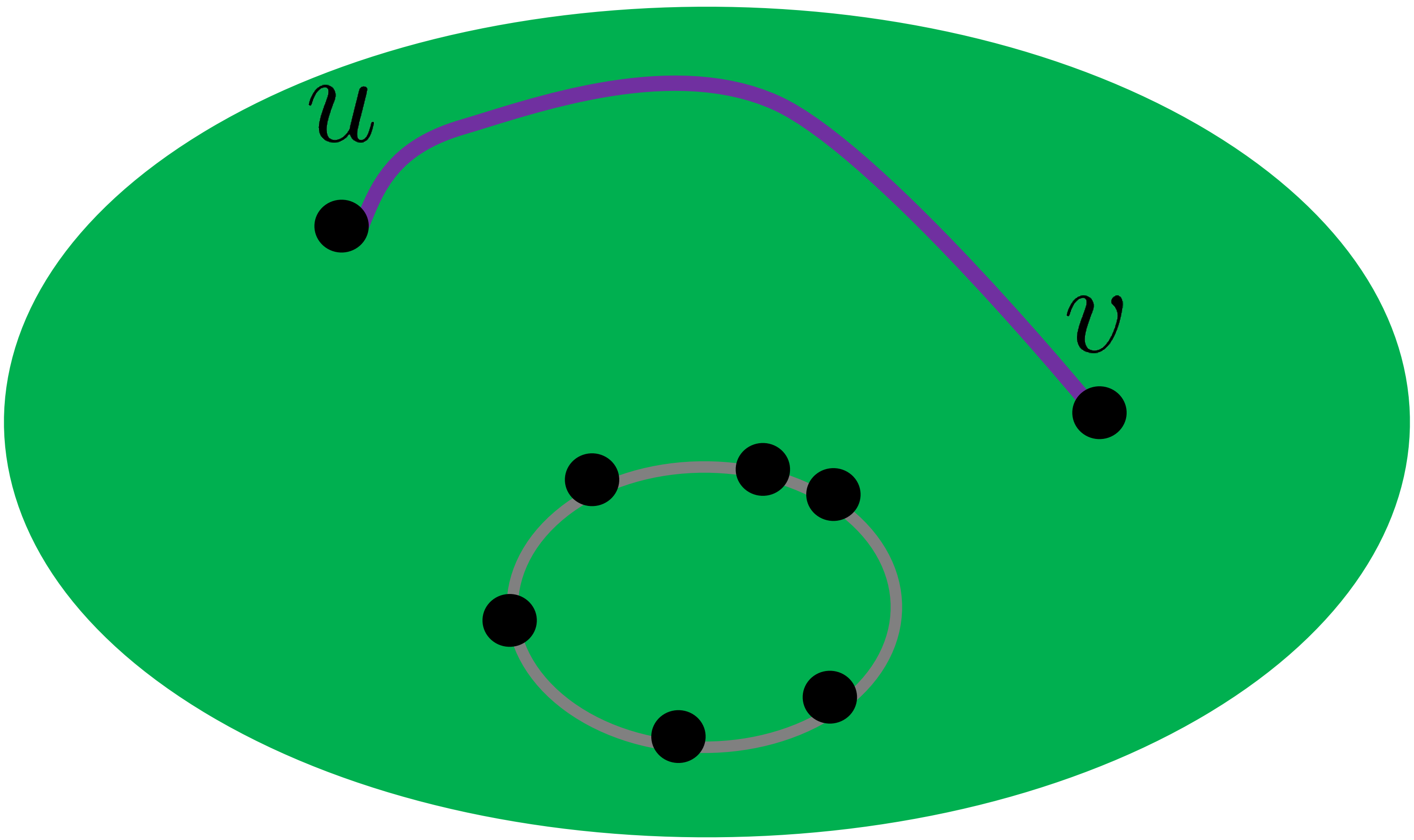
Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

u, v query:

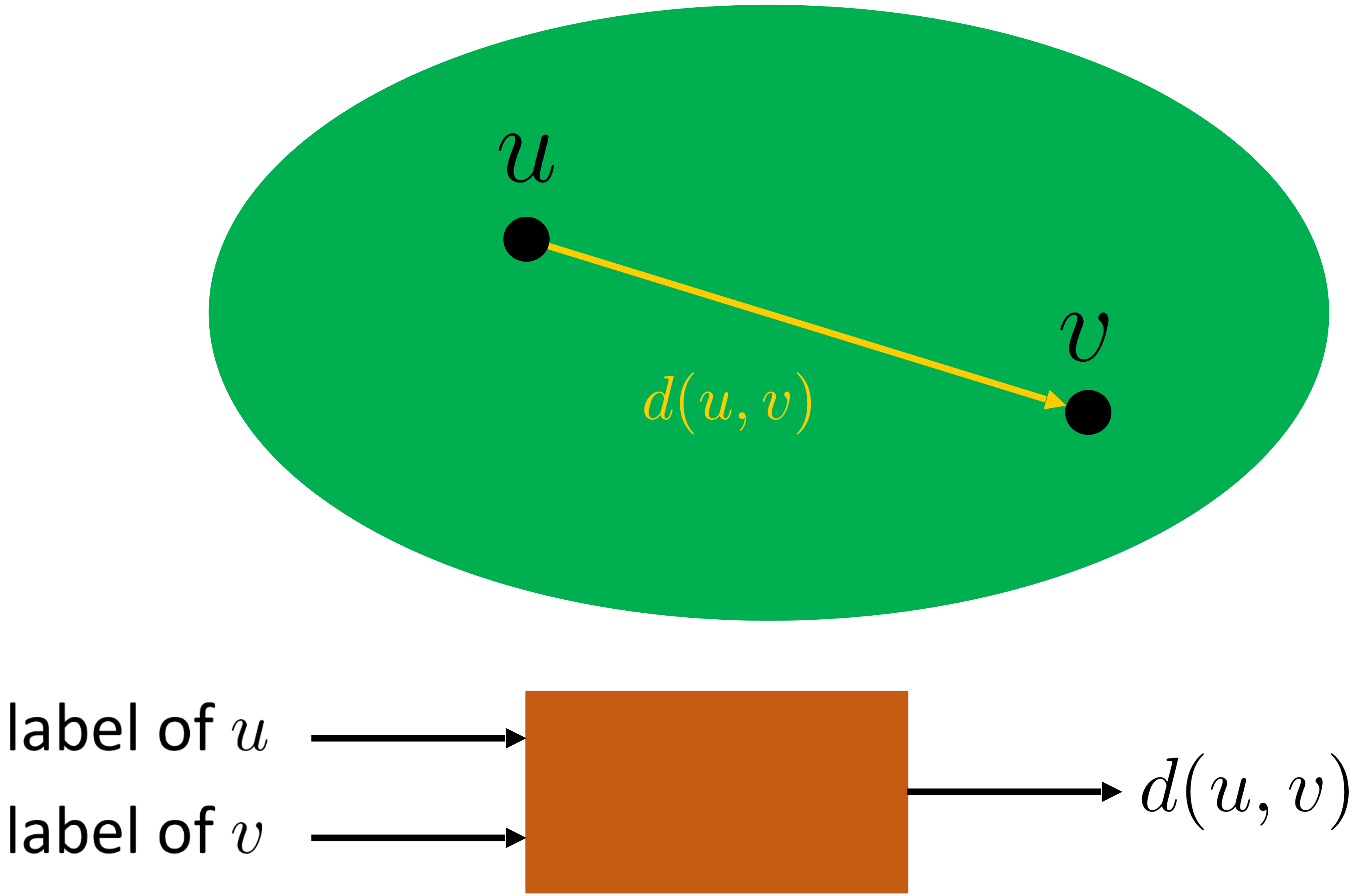


Example: $O(\sqrt{n} \cdot \log n)$ distance labeling

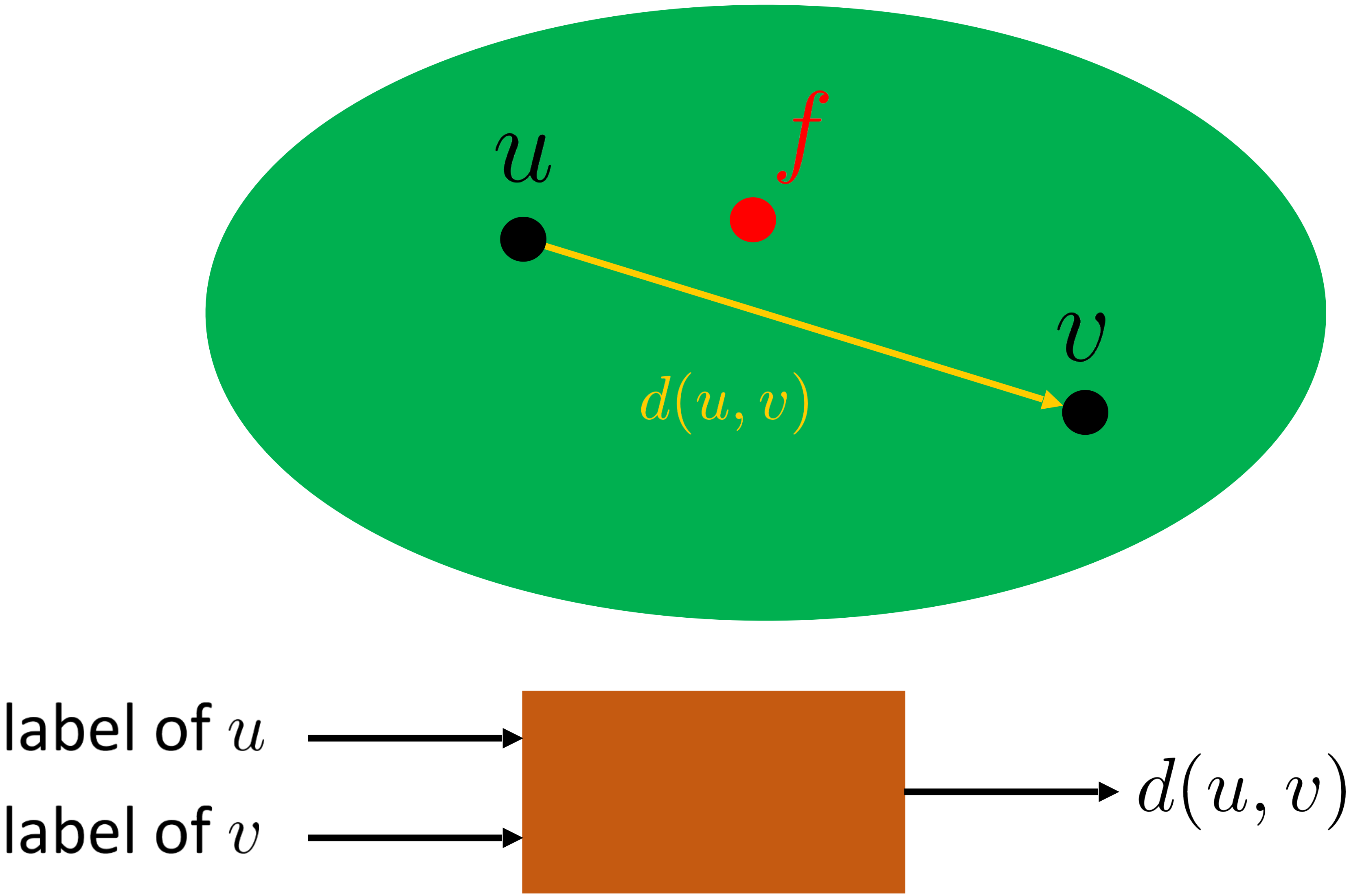
u, v query:



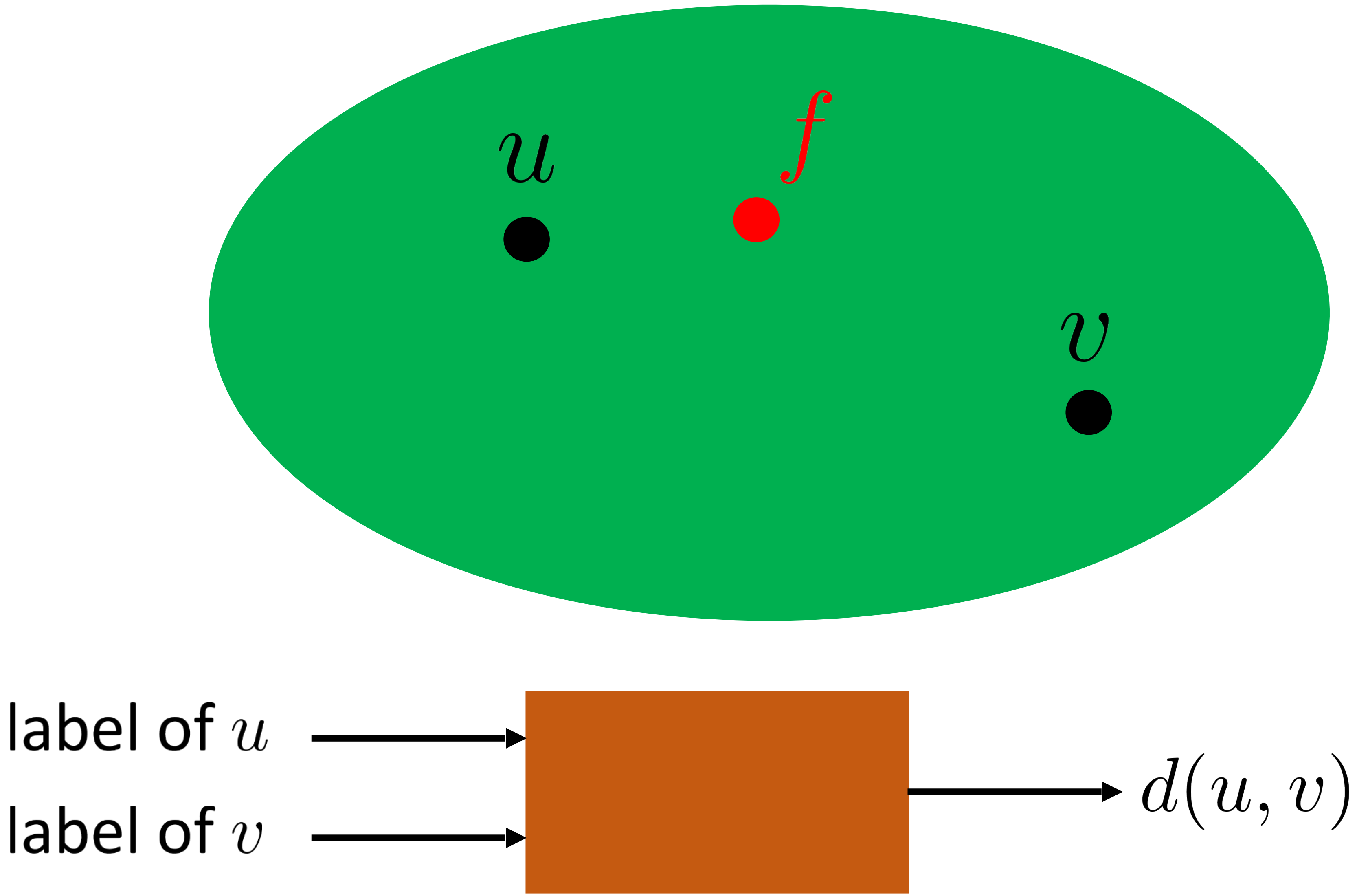
Fault-Tolerant Setting



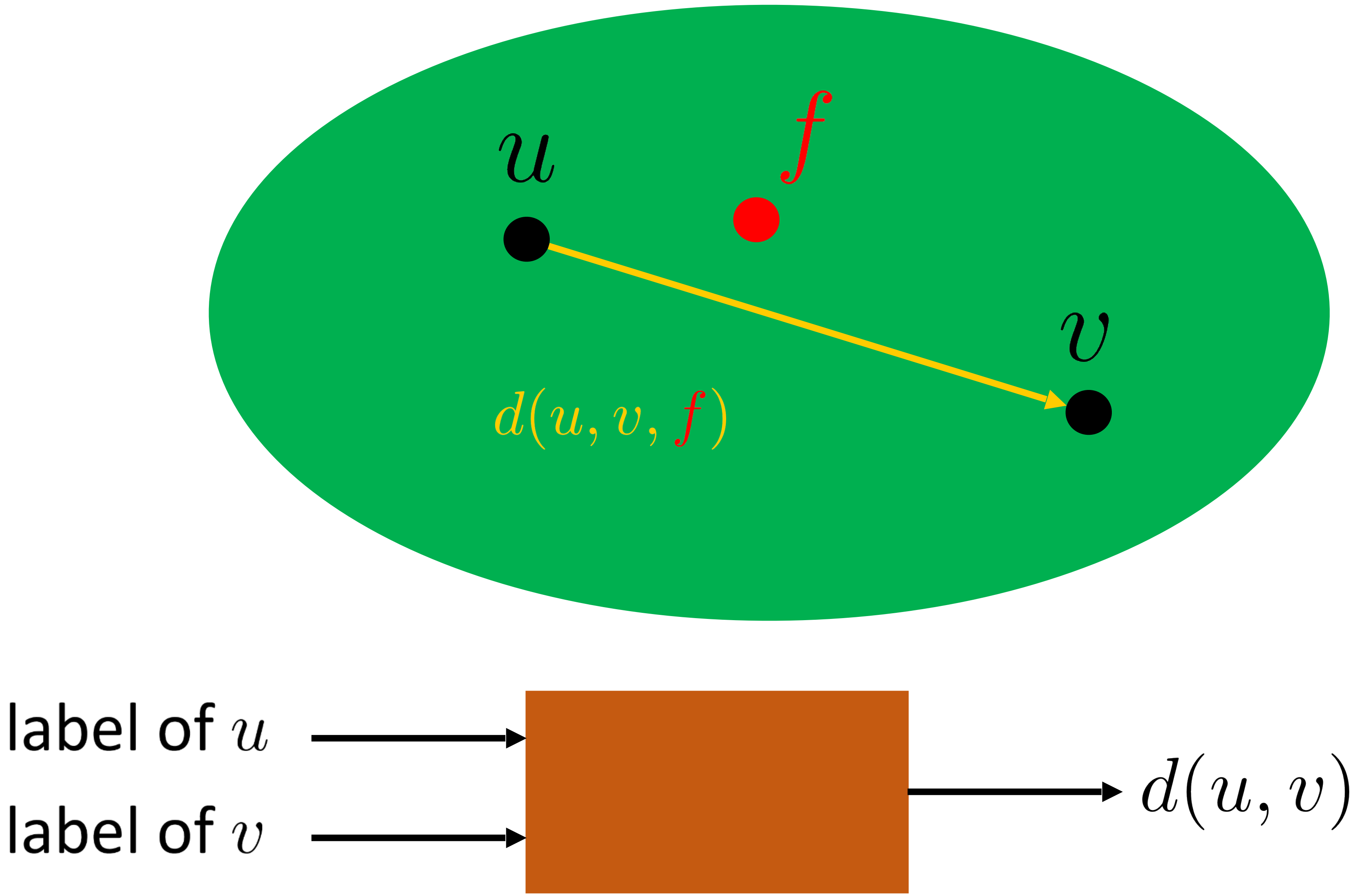
Fault-Tolerant Setting



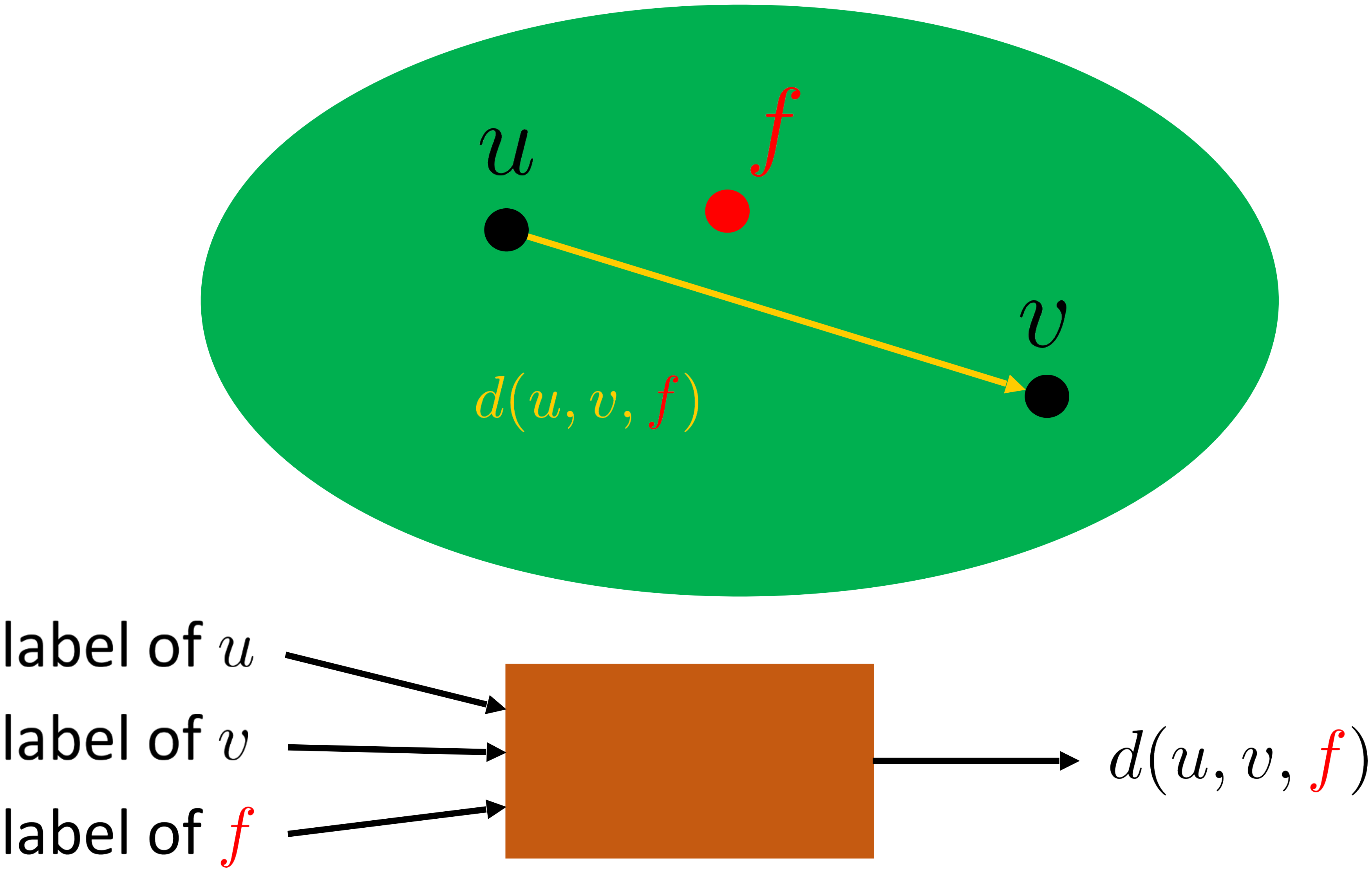
Fault-Tolerant Setting



Fault-Tolerant Setting



Fault-Tolerant Setting

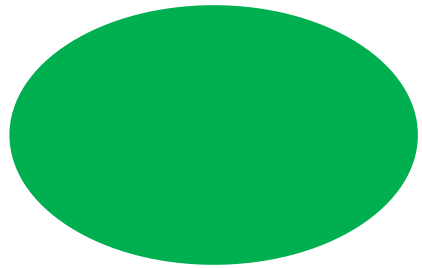


Our result

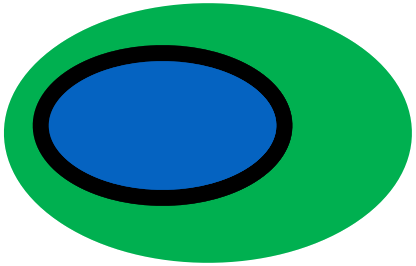
Labels of size $\tilde{O}(n^{2/3})$ for fault-tolerant distance labeling in planar graphs

Recursive decomposition tree

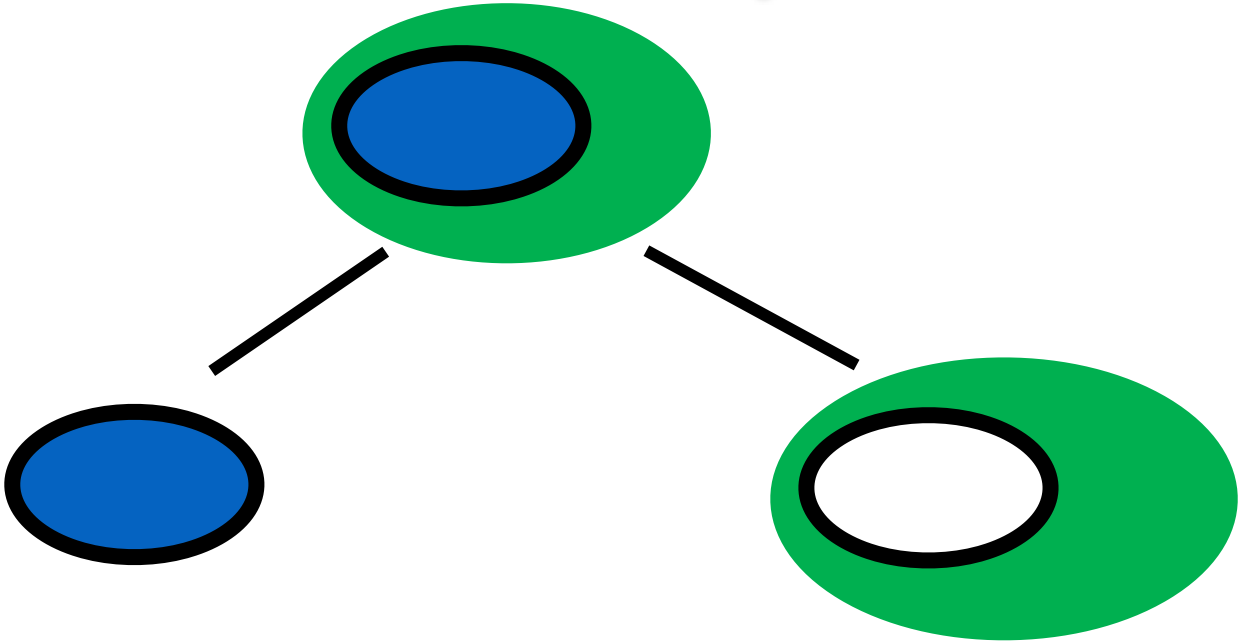
Recursive decomposition tree



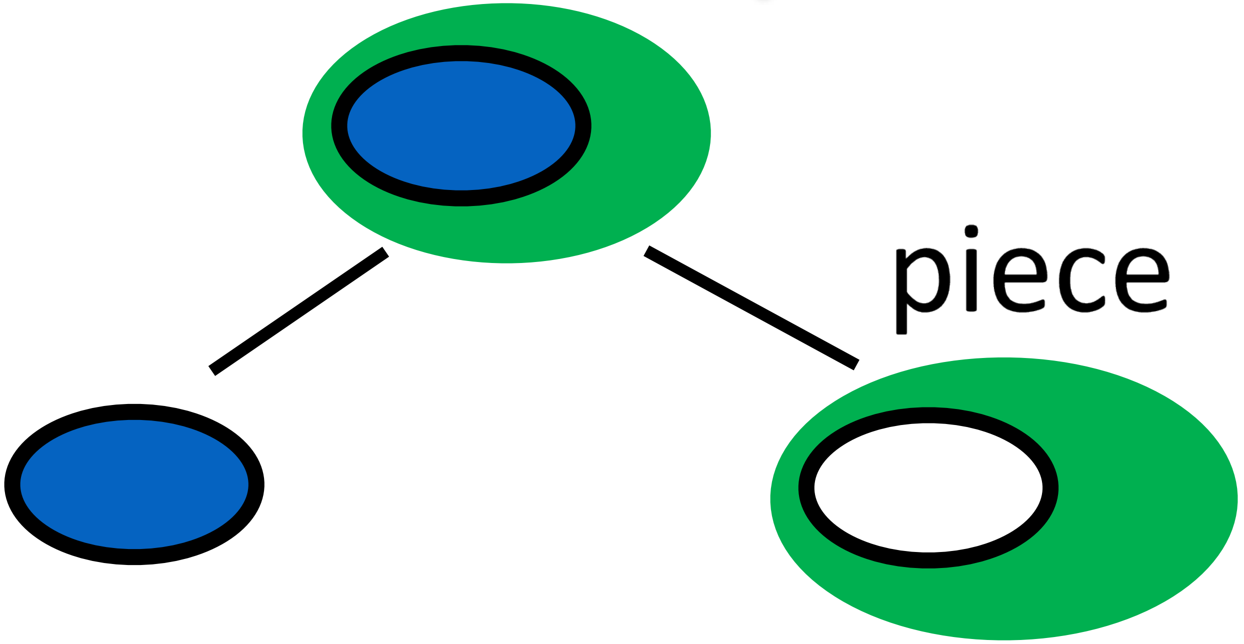
Recursive decomposition tree



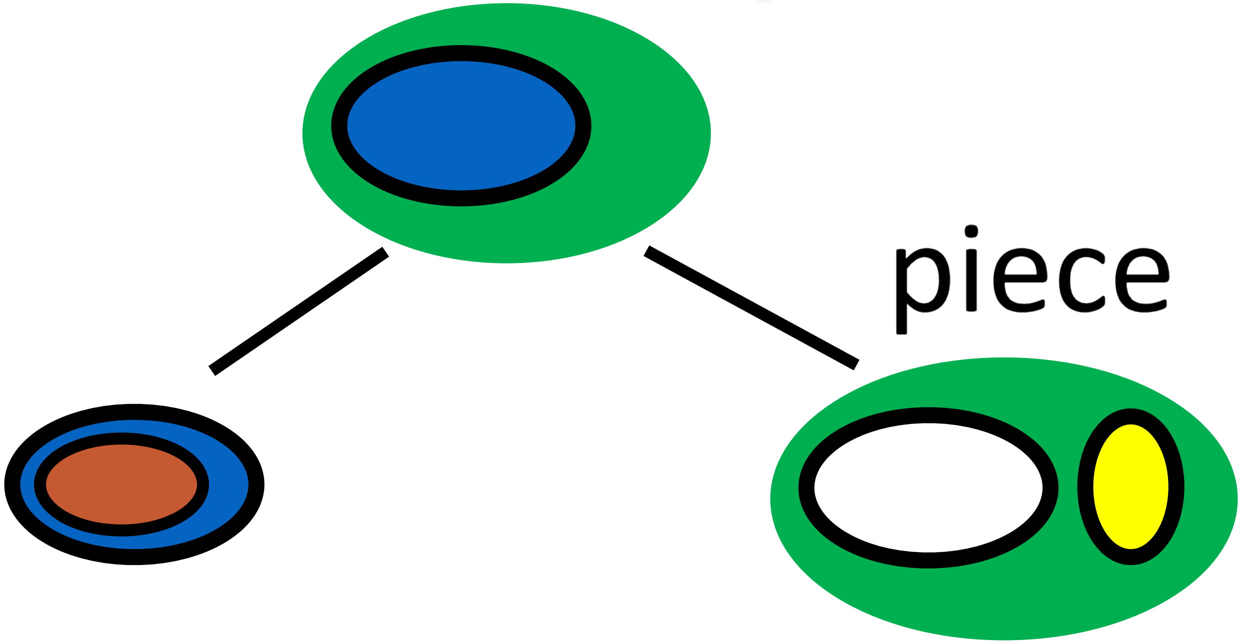
Recursive decomposition tree



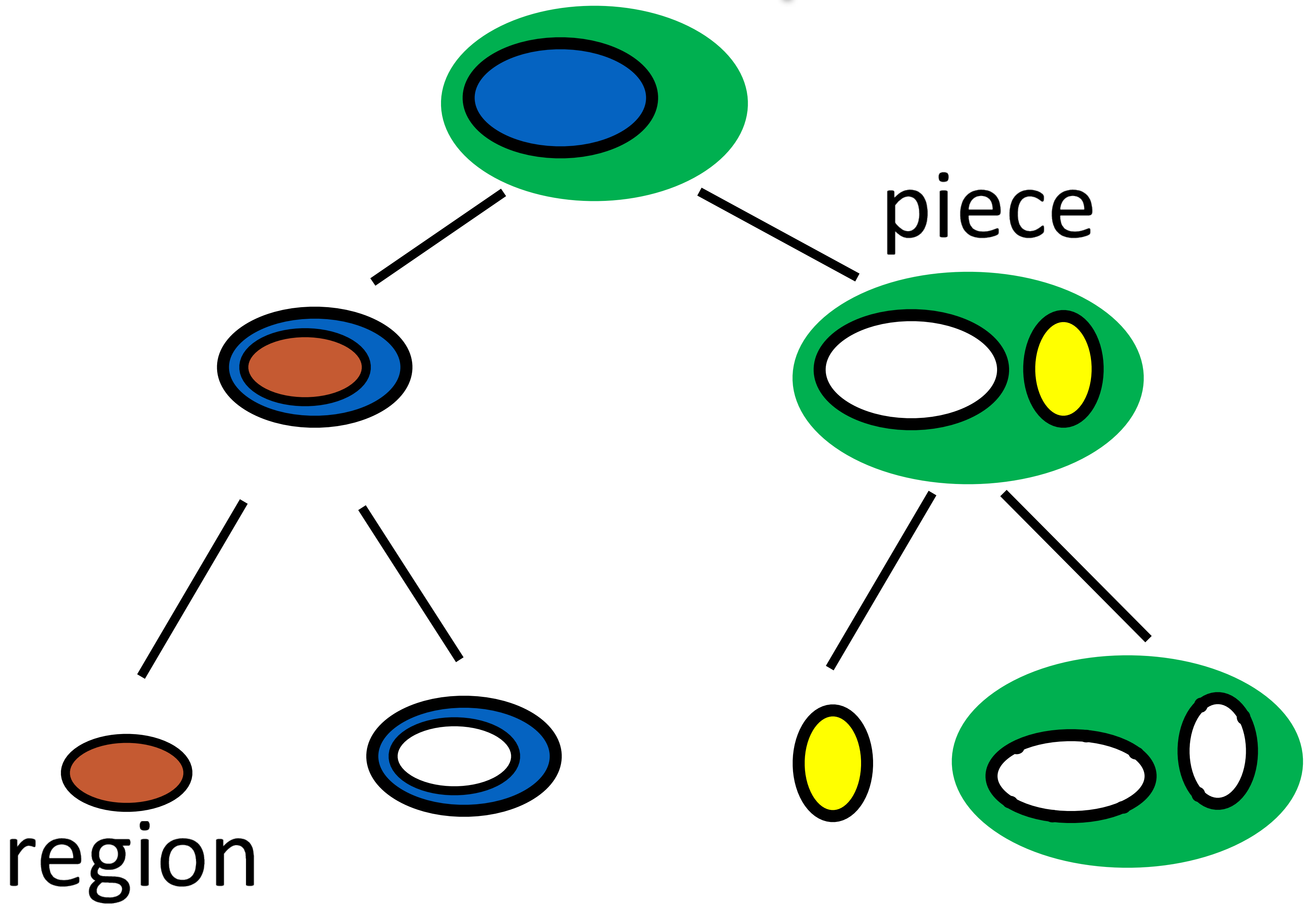
Recursive decomposition tree



Recursive decomposition tree

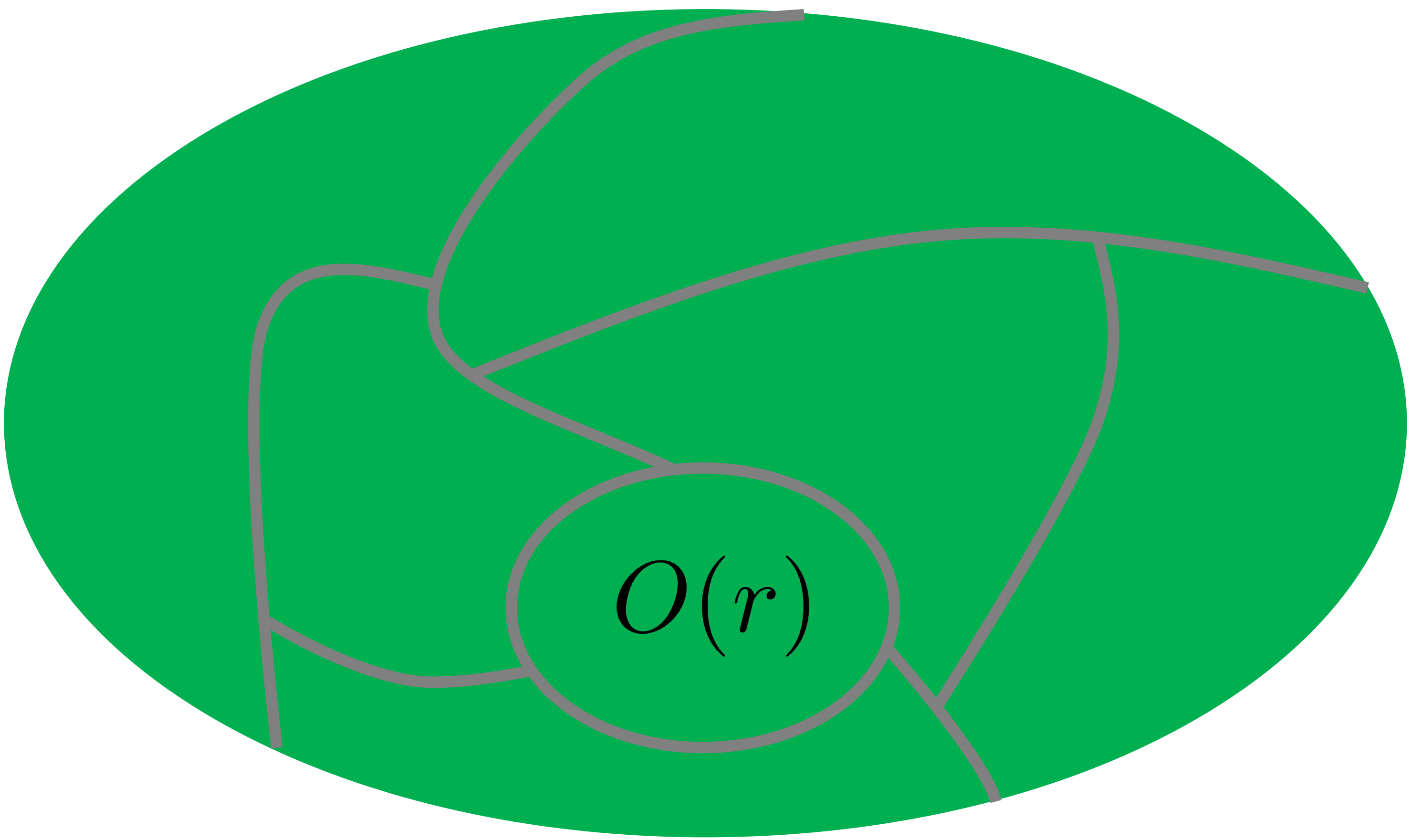


Recursive decomposition tree



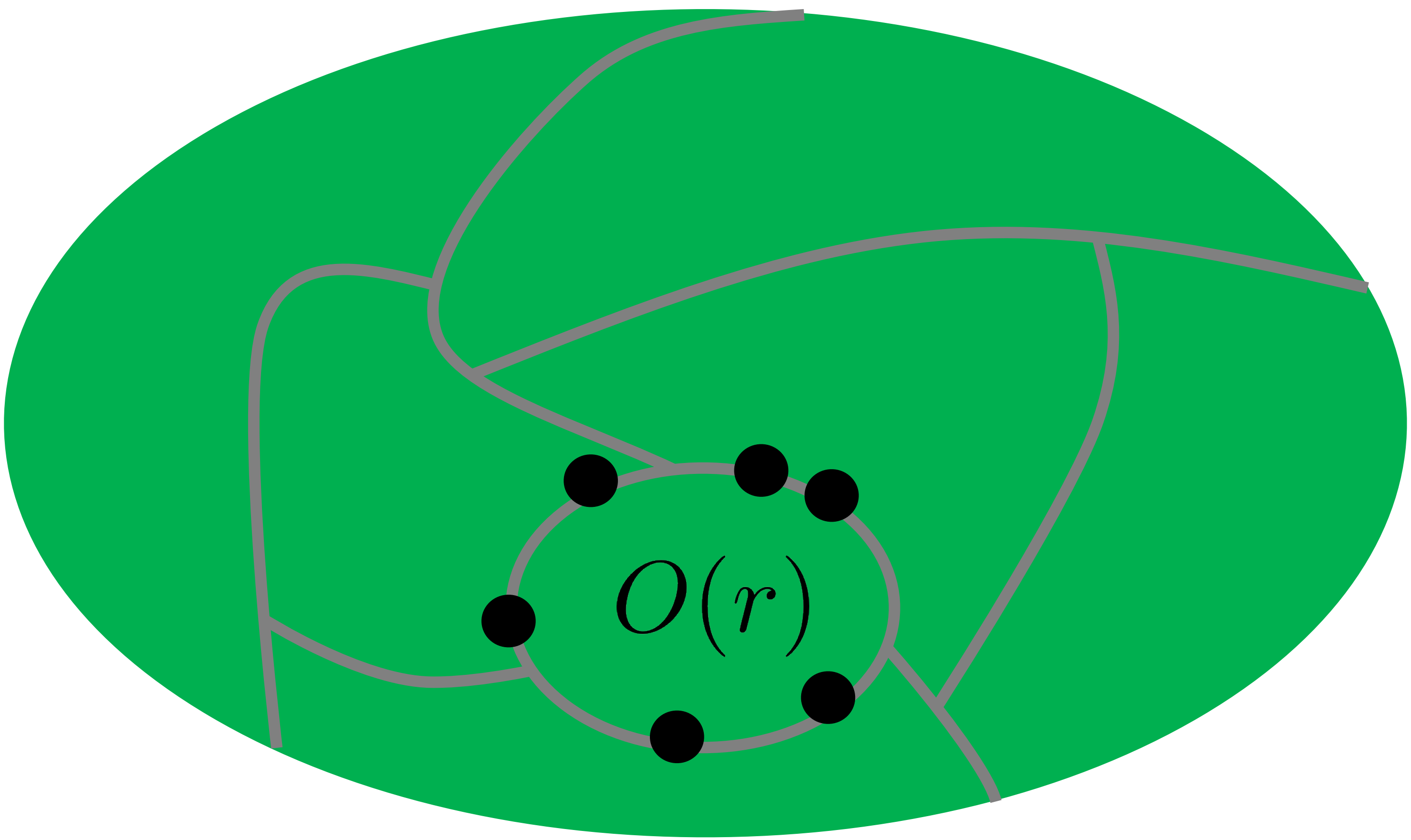
r-division

$O(n/r)$ regions, each with $O(r)$ vertices and $O(\sqrt{r})$ boundary vertices



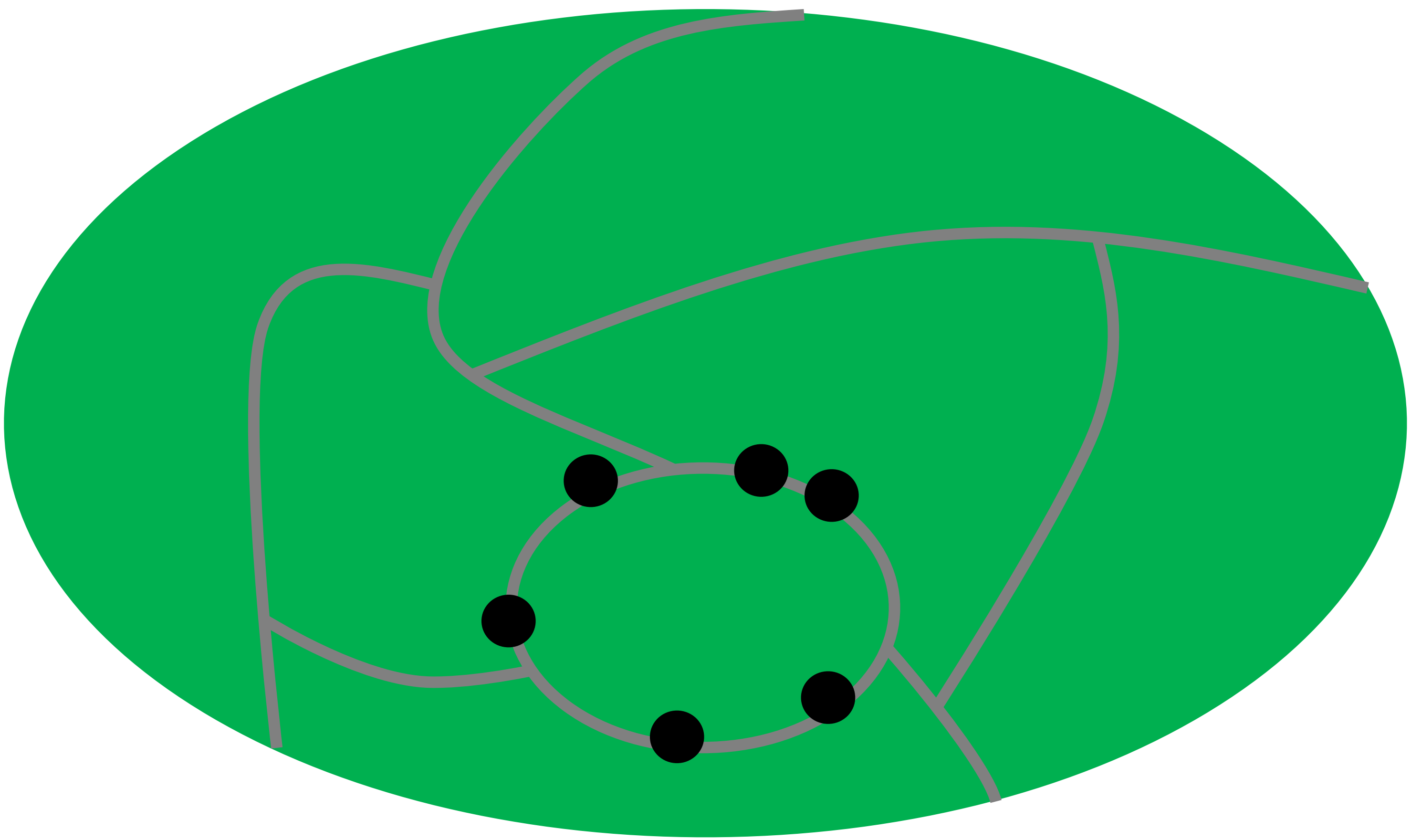
r-division

$O(n/r)$ regions, each with $O(r)$ vertices and $O(\sqrt{r})$ boundary vertices



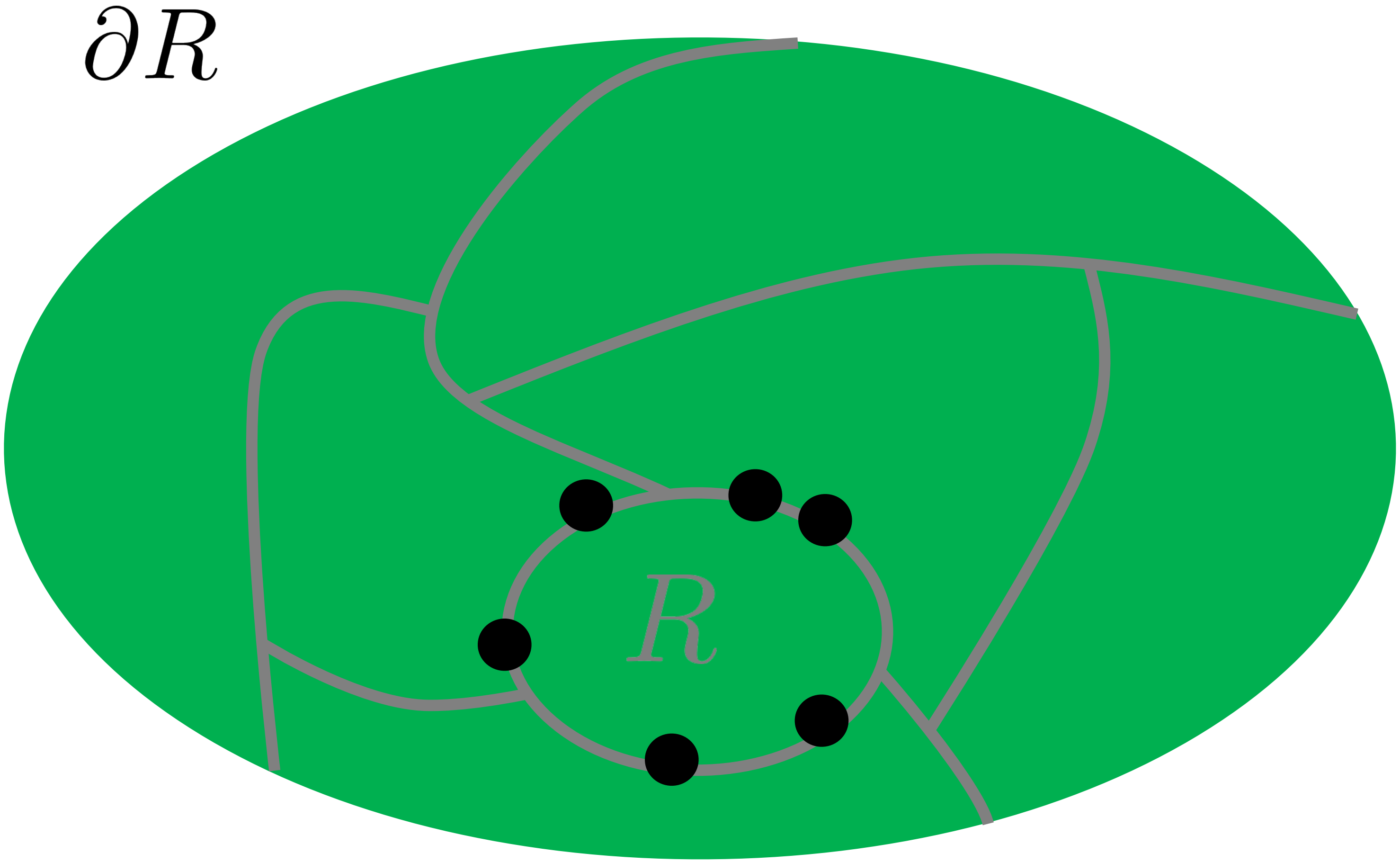
r-division

$O(n/r)$ regions, each with $O(r)$ vertices and $O(\sqrt{r})$ boundary vertices



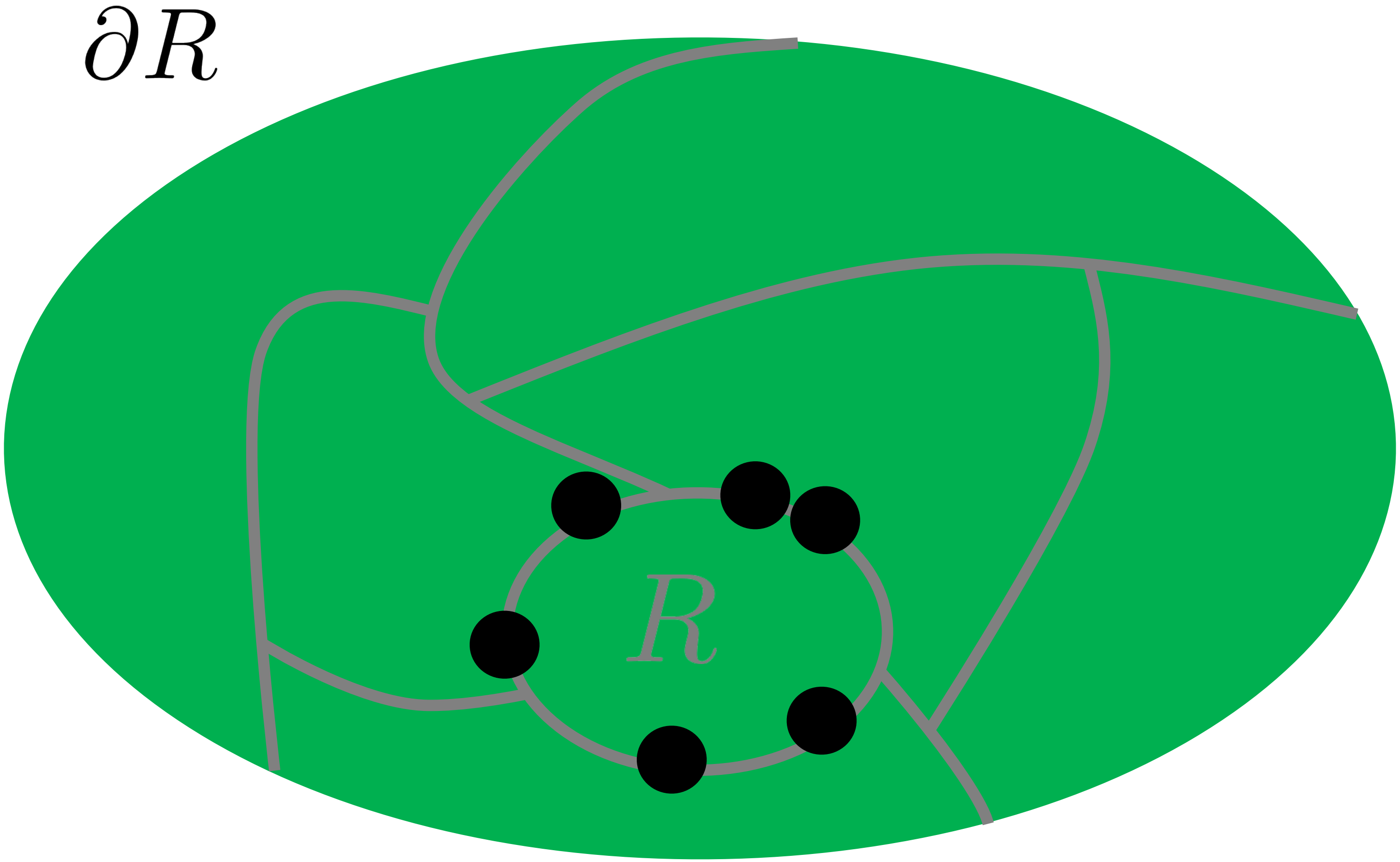
r-division

$O(n/r)$ regions, each with $O(r)$ vertices and $O(\sqrt{r})$ boundary vertices

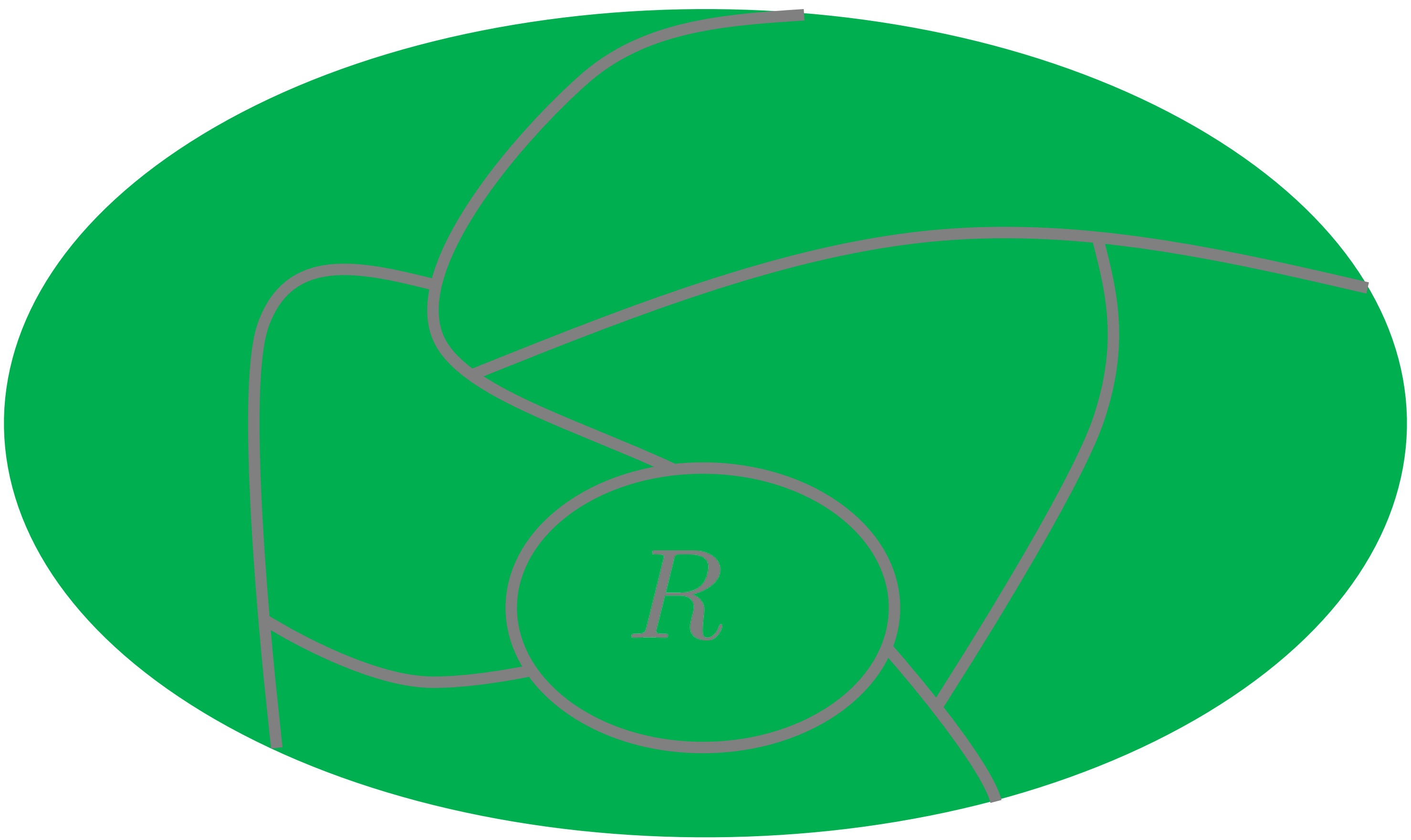


r-division

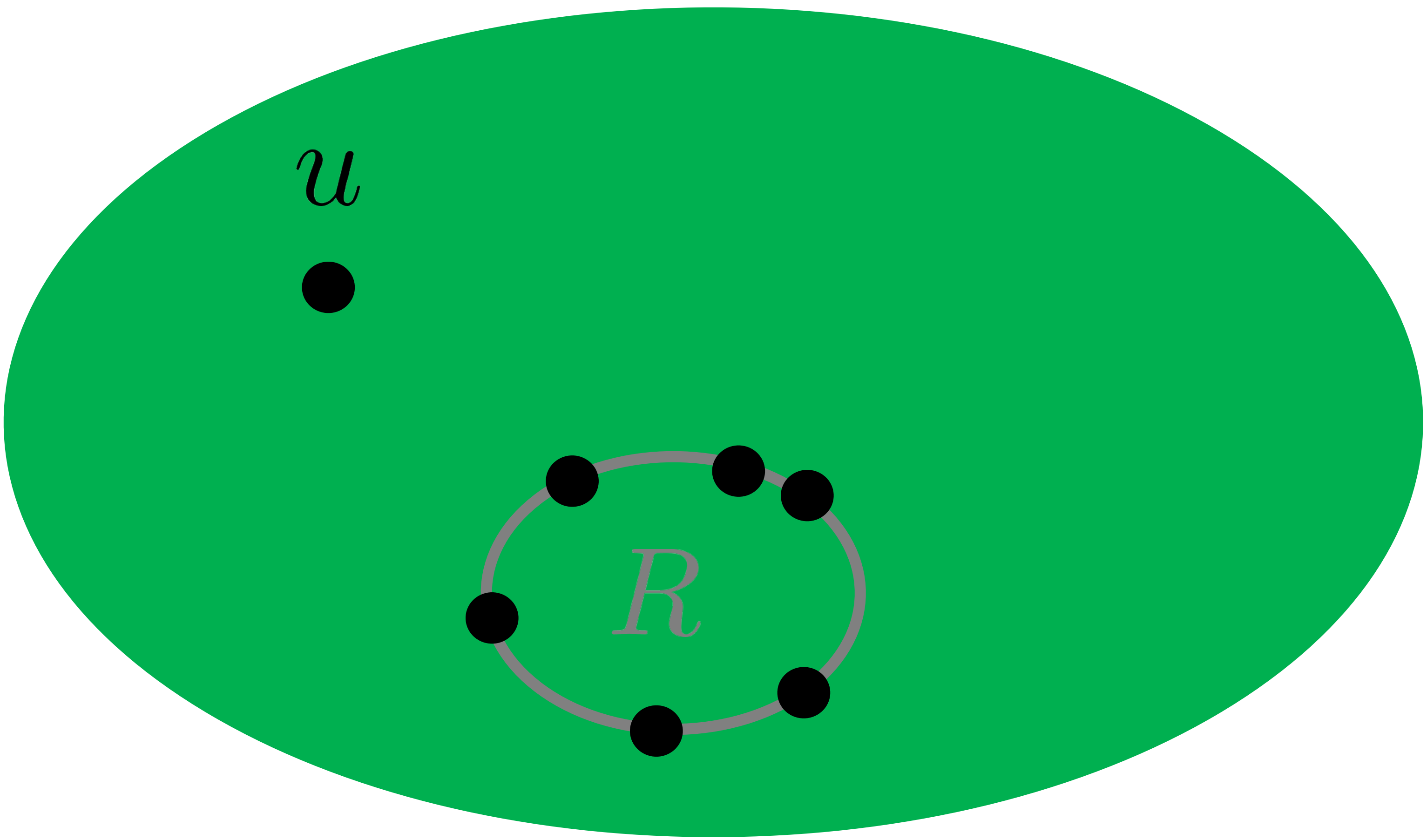
$O(n/r)$ regions, each with $O(r)$ vertices and $O(\sqrt{r})$ boundary vertices



Fault tolerant distance labels

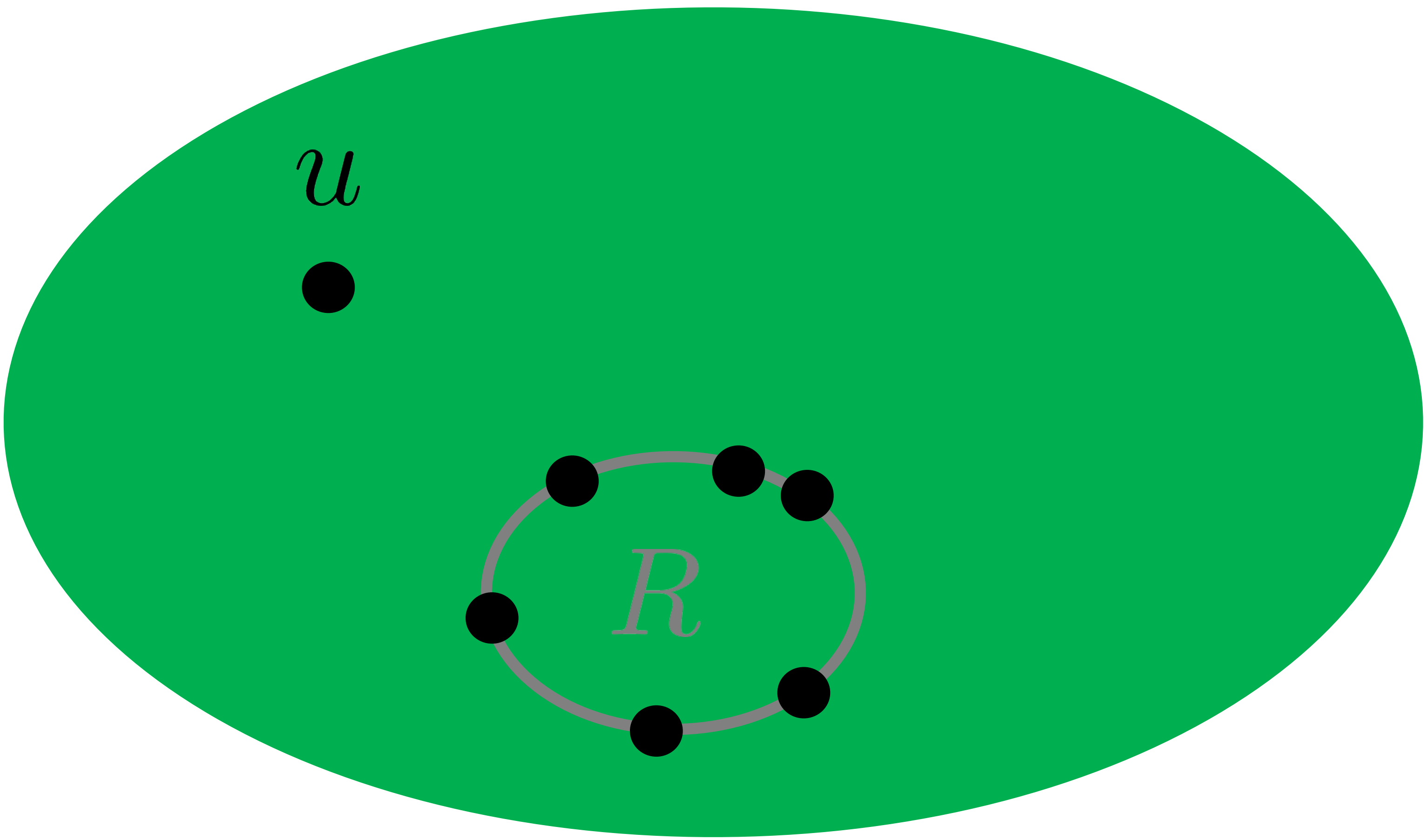


u's label:



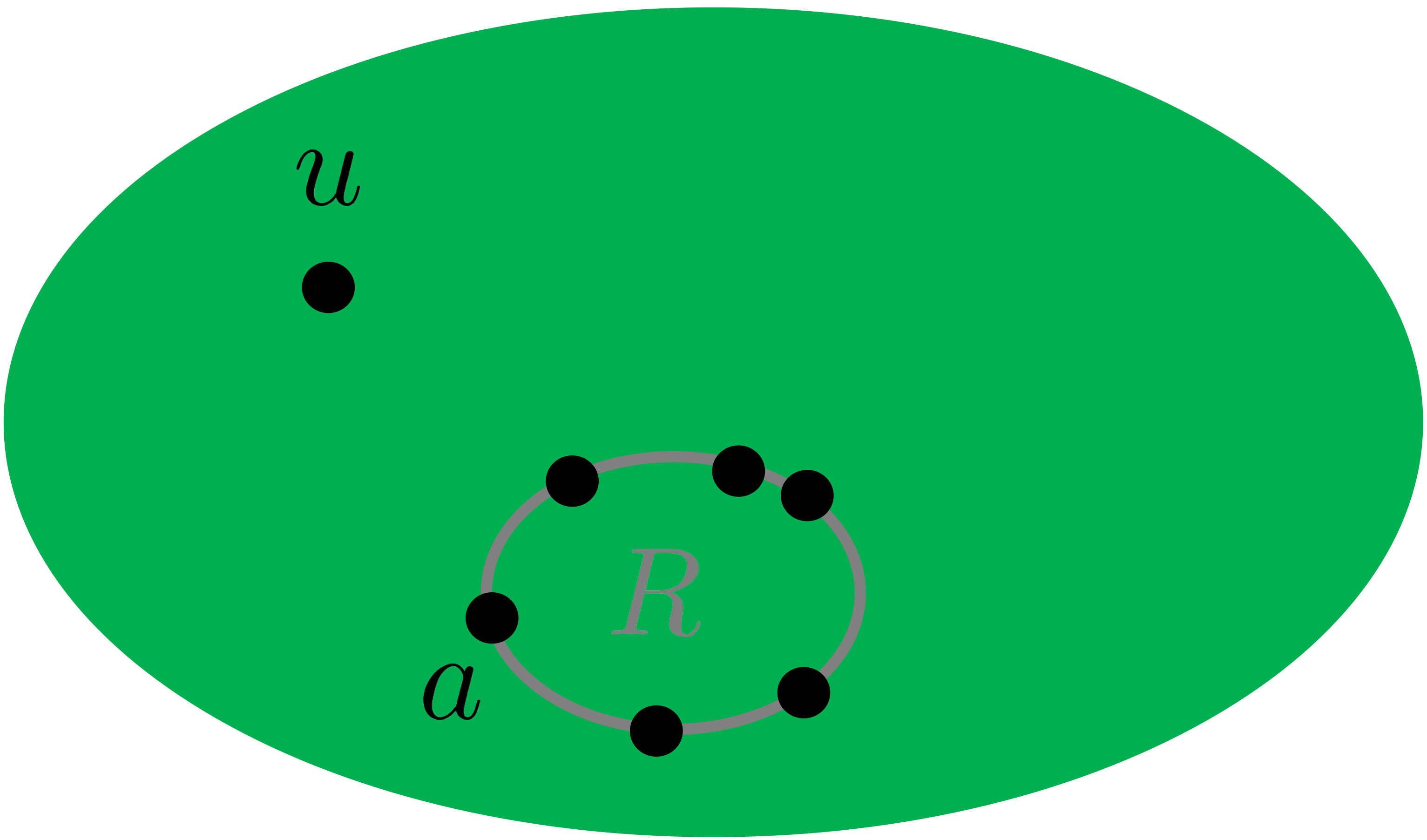
u's label:

u to ∂R internally disjoint from R for every region R



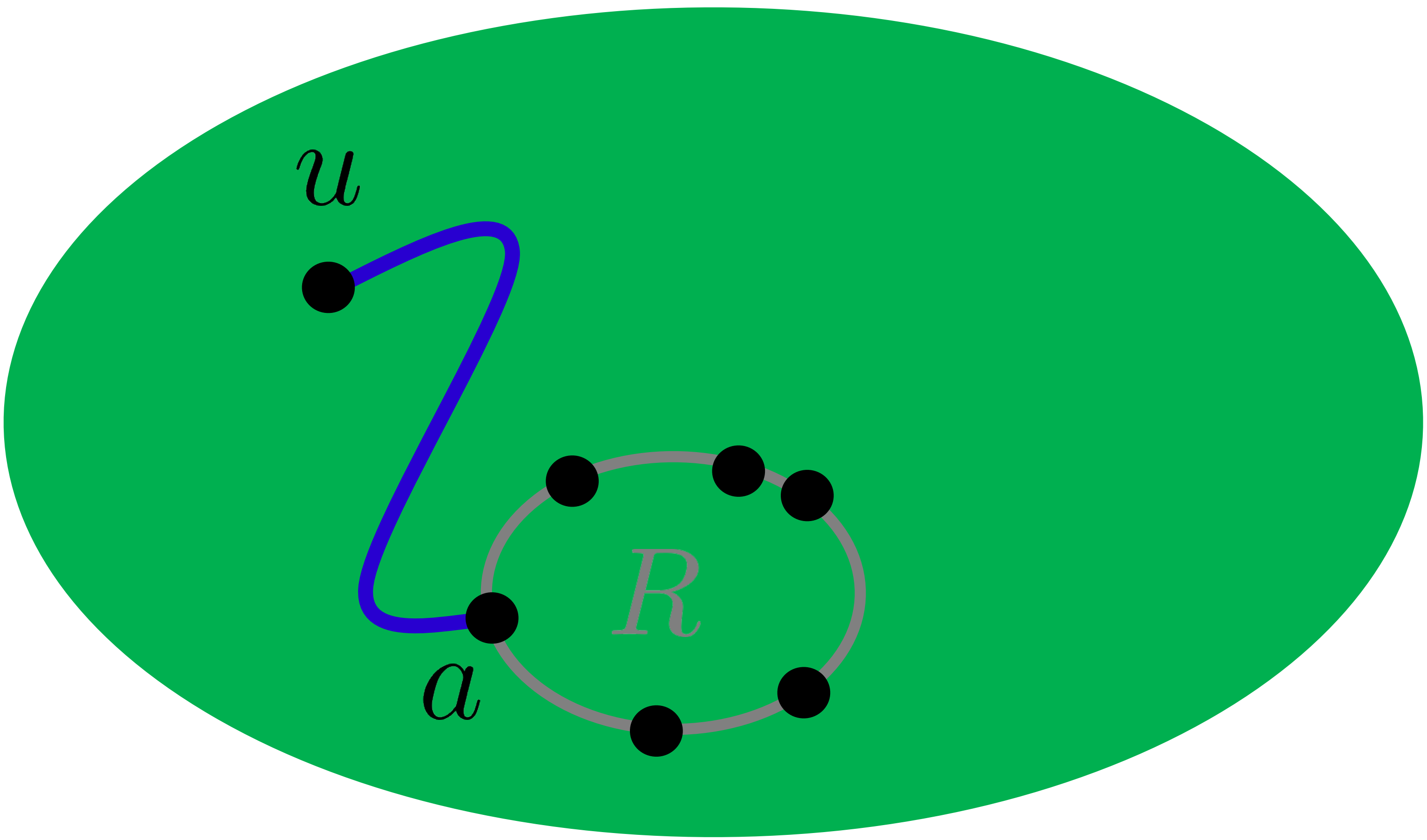
u's label:

u to ∂R internally disjoint from R for every region R



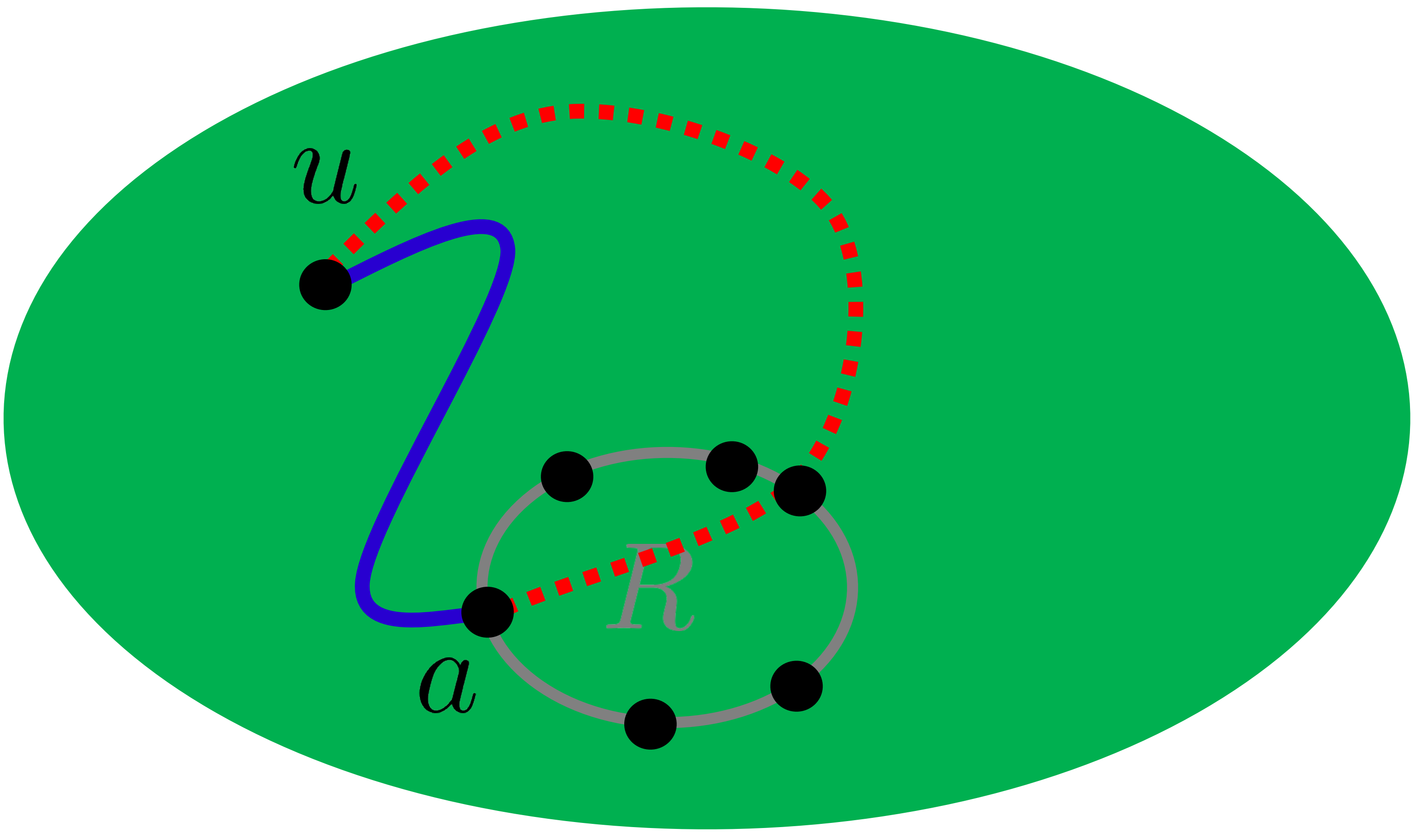
u's label:

u to ∂R internally disjoint from R for every region R



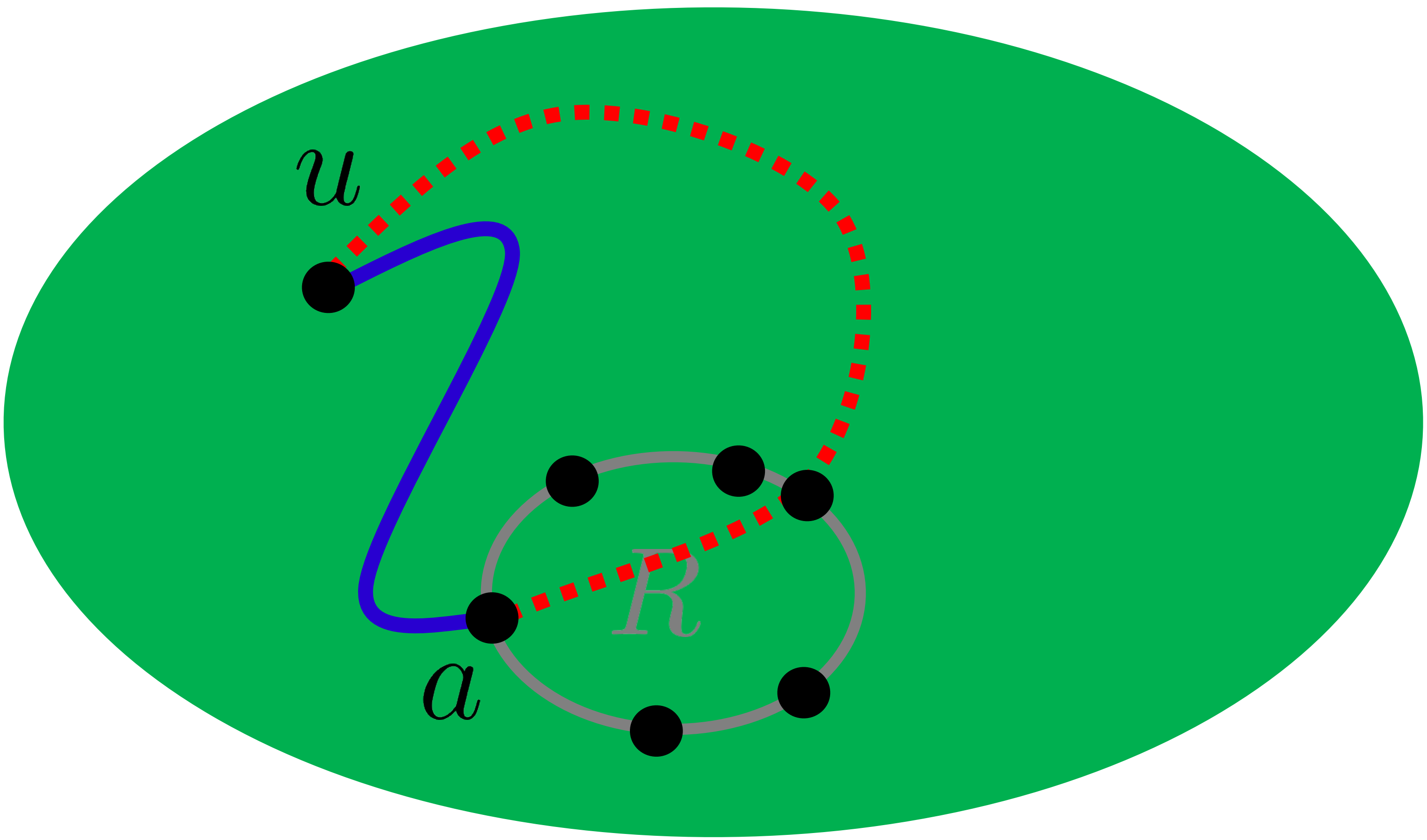
u's label:

u to ∂R internally disjoint from R for every region R



u's label:

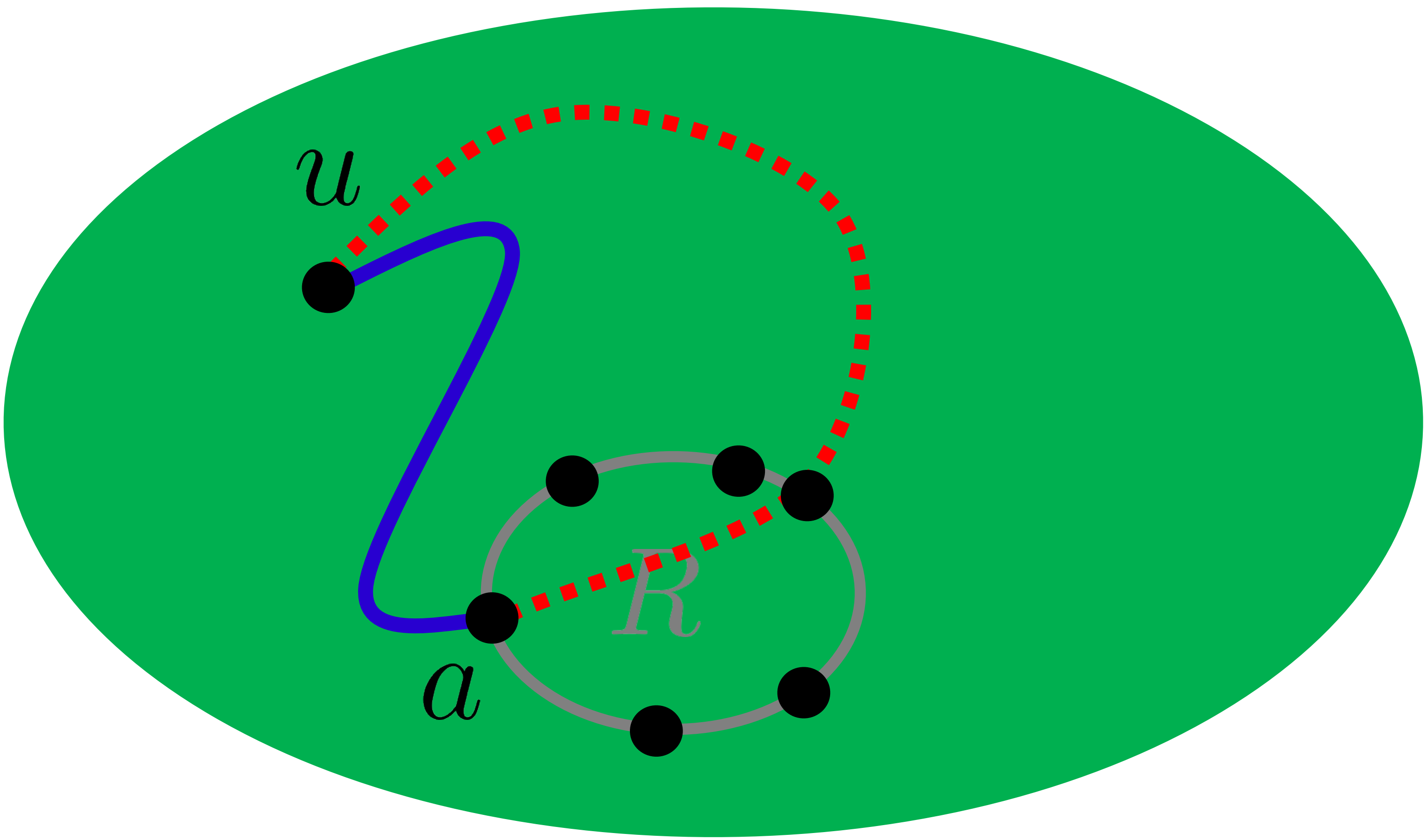
u to ∂R internally disjoint from R for every region R



u's label:

u to ∂R internally disjoint from R for every region R

$\text{space} = \#regions \cdot |\partial R|$

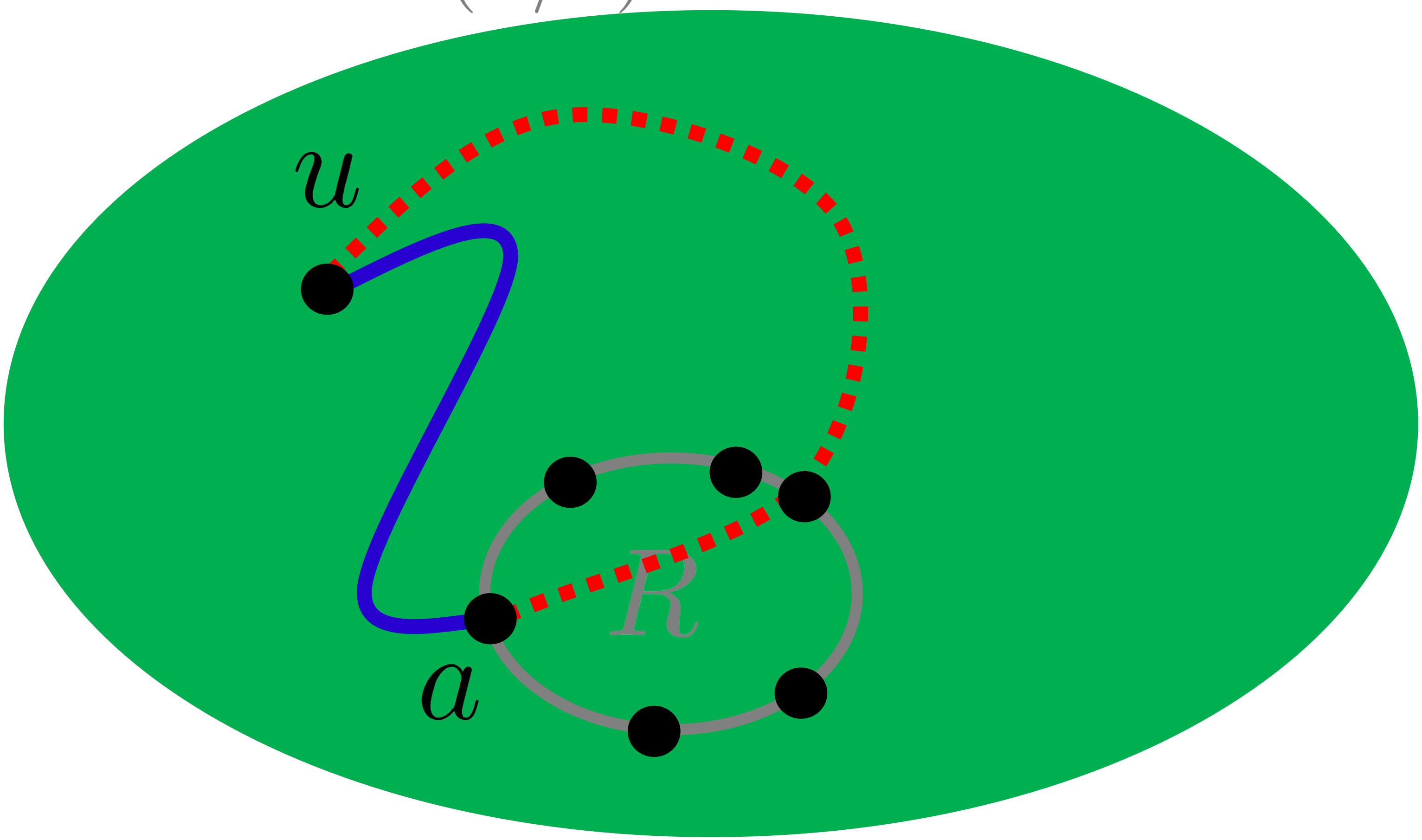


u's label:

u to ∂R internally disjoint from R for every region R

$$\text{space} = \#regions \cdot |\partial R|$$

$O(n/r)$

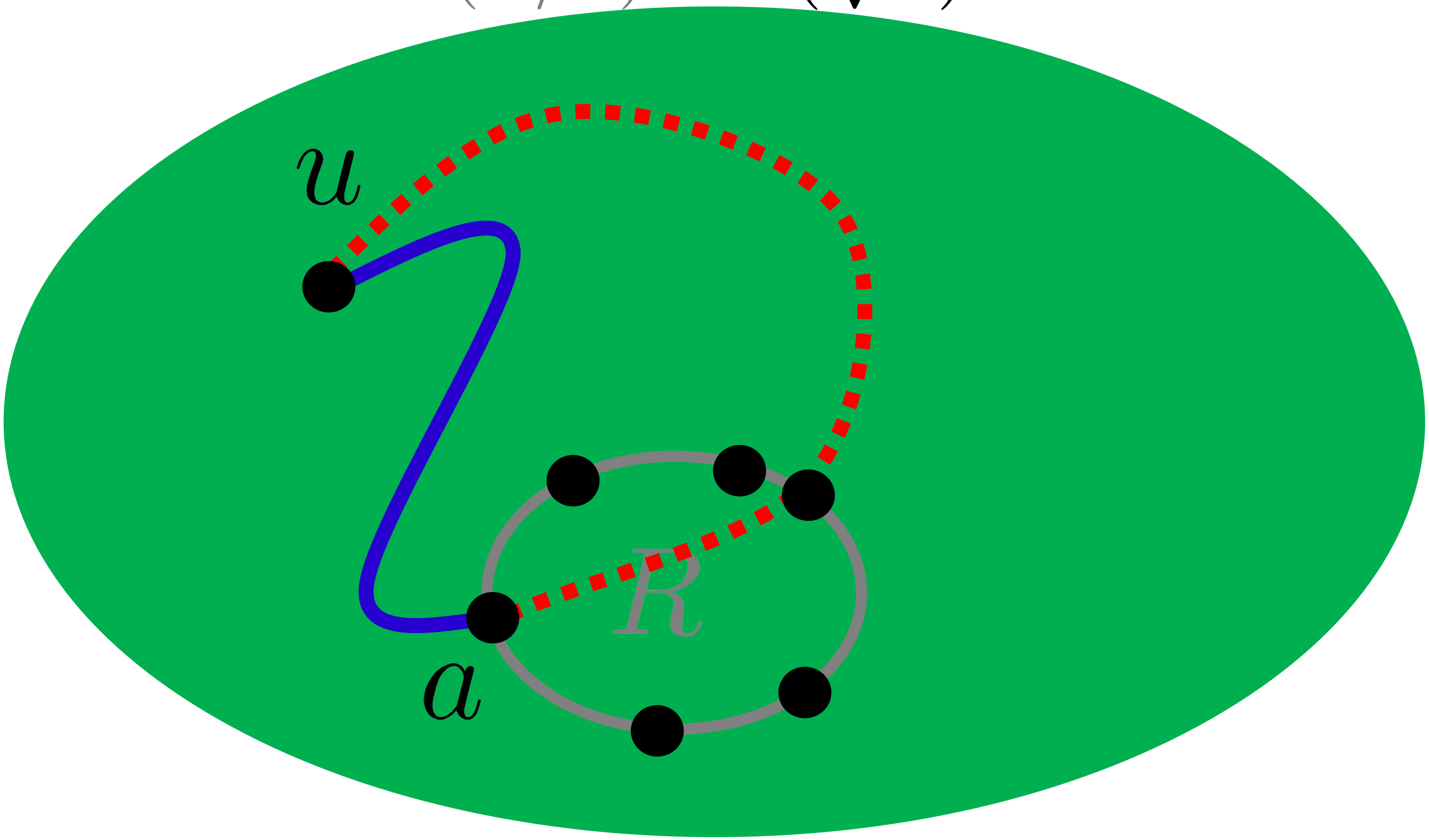


u's label:

u to ∂R internally disjoint from R for every region R

$$\text{space} = \#regions \cdot |\partial R|$$

$$O(n/r) \cdot O(\sqrt{r})$$

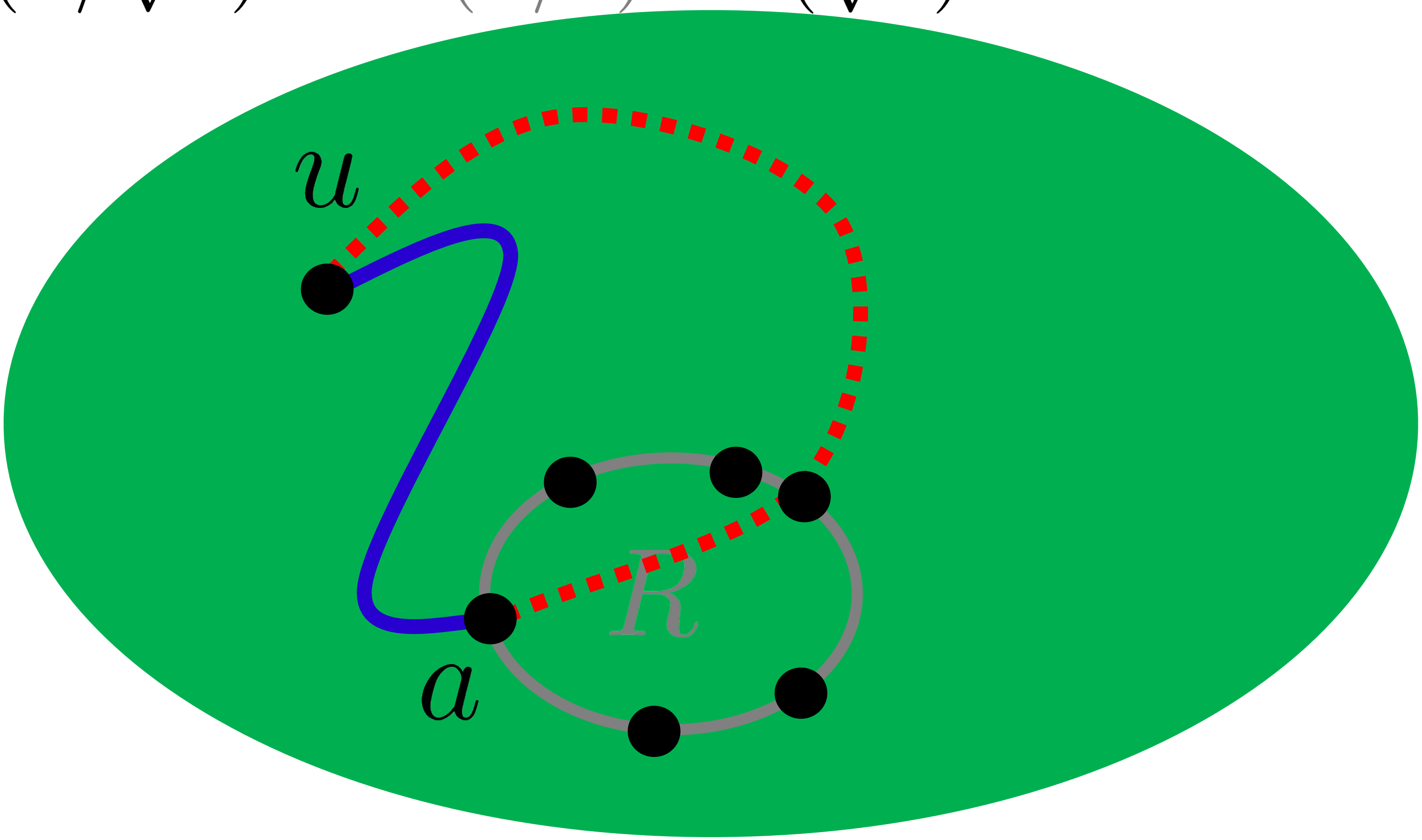


u's label:

u to ∂R internally disjoint from R for every region R

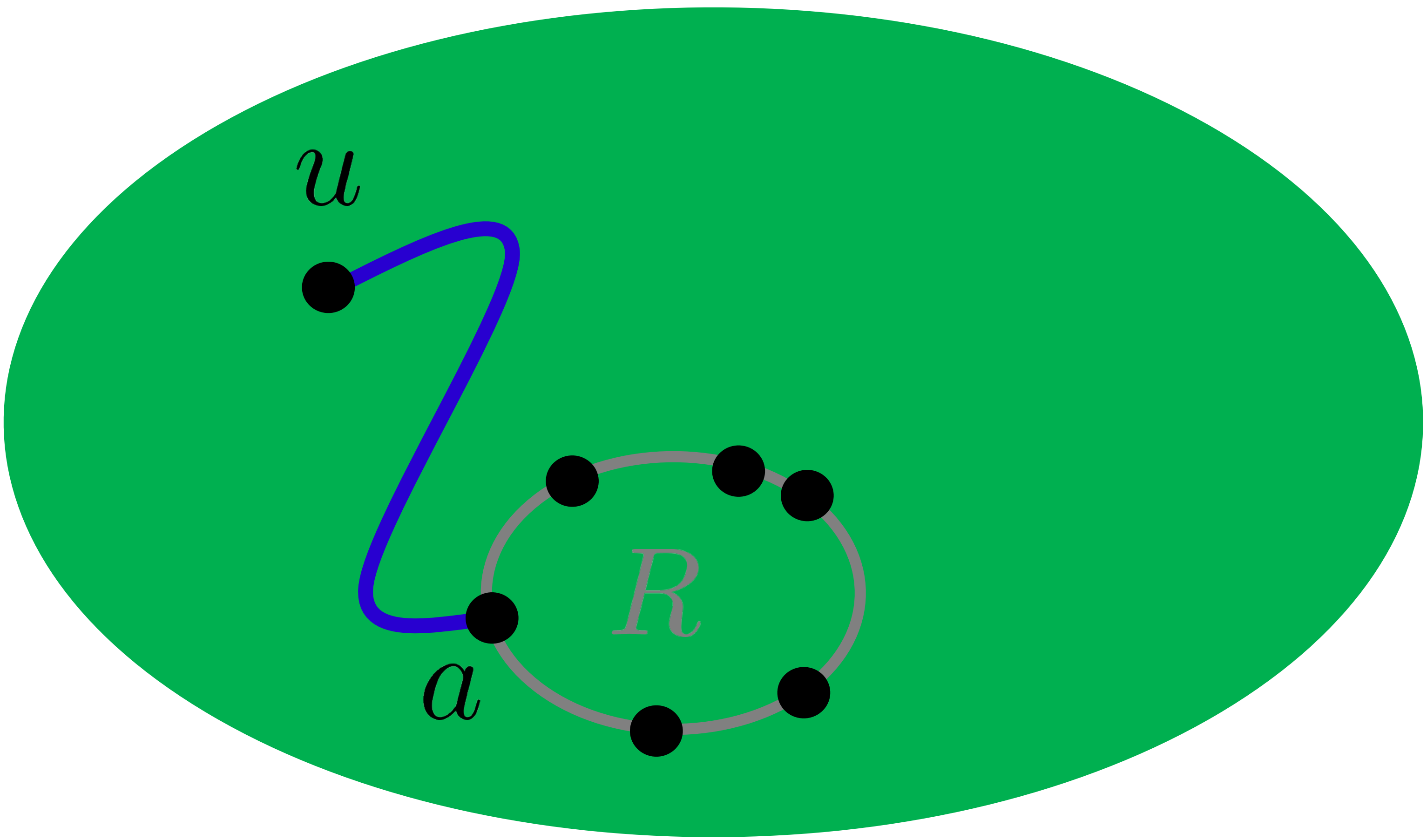
$$\text{space} = \#regions \cdot |\partial R|$$

$$\tilde{O}(n/\sqrt{r}) = O(n/r) \cdot O(\sqrt{r})$$



u's label:

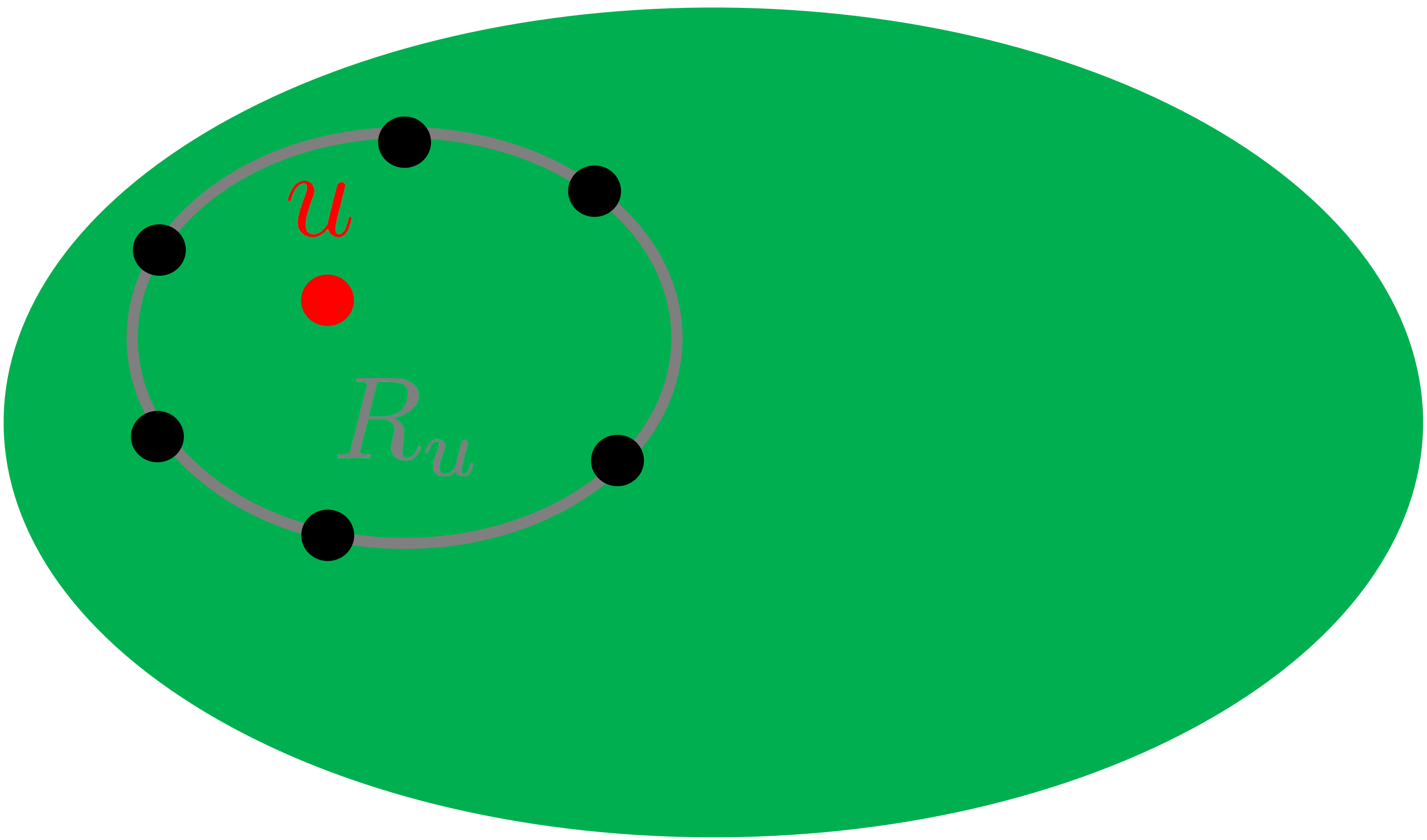
u to ∂R internally disjoint from R for every region R



u's label:

u to ∂R internally disjoint from R for every region R

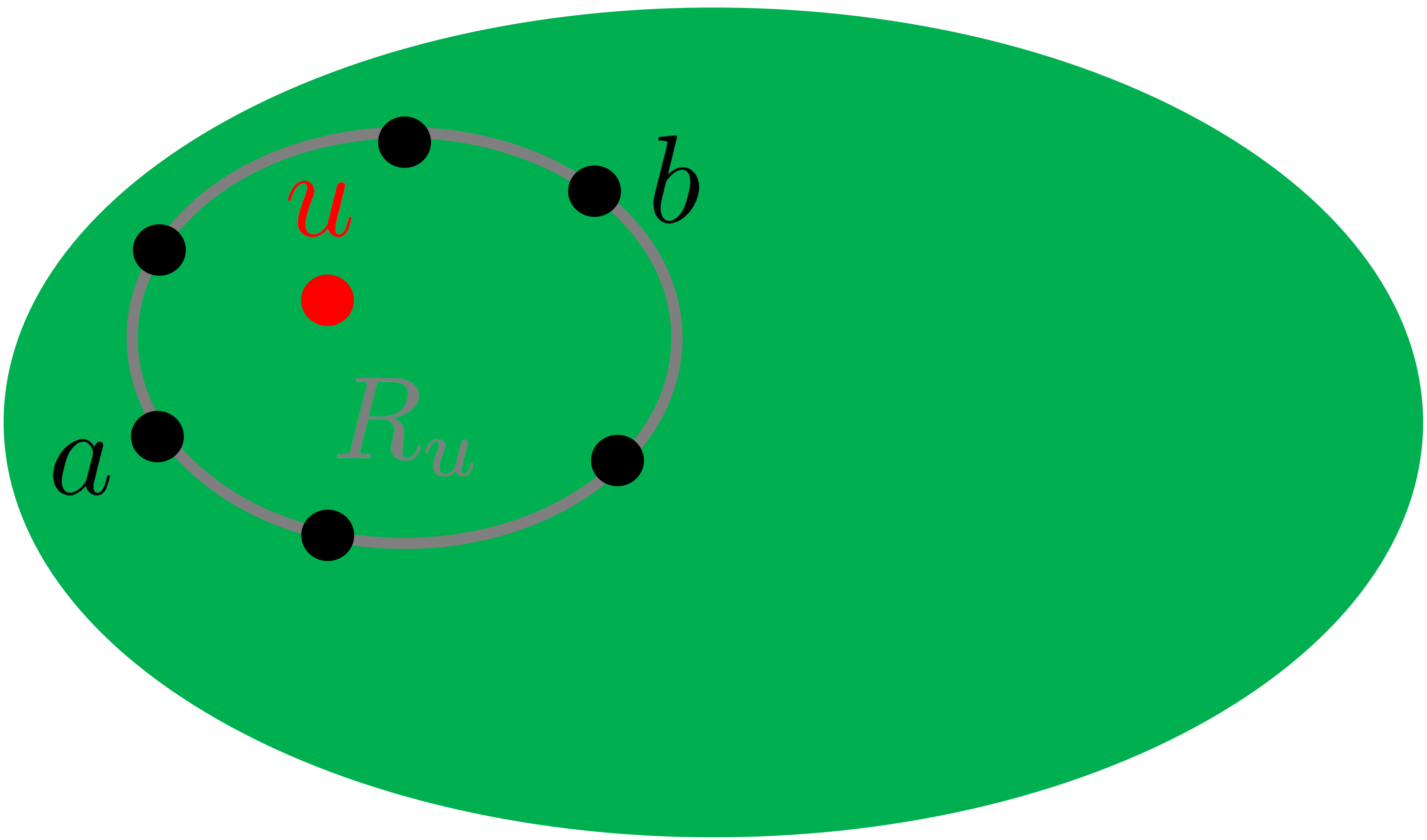
∂R_u to ∂R_u in $G \backslash \{u\}$



u's label:

u to ∂R internally disjoint from R for every region R

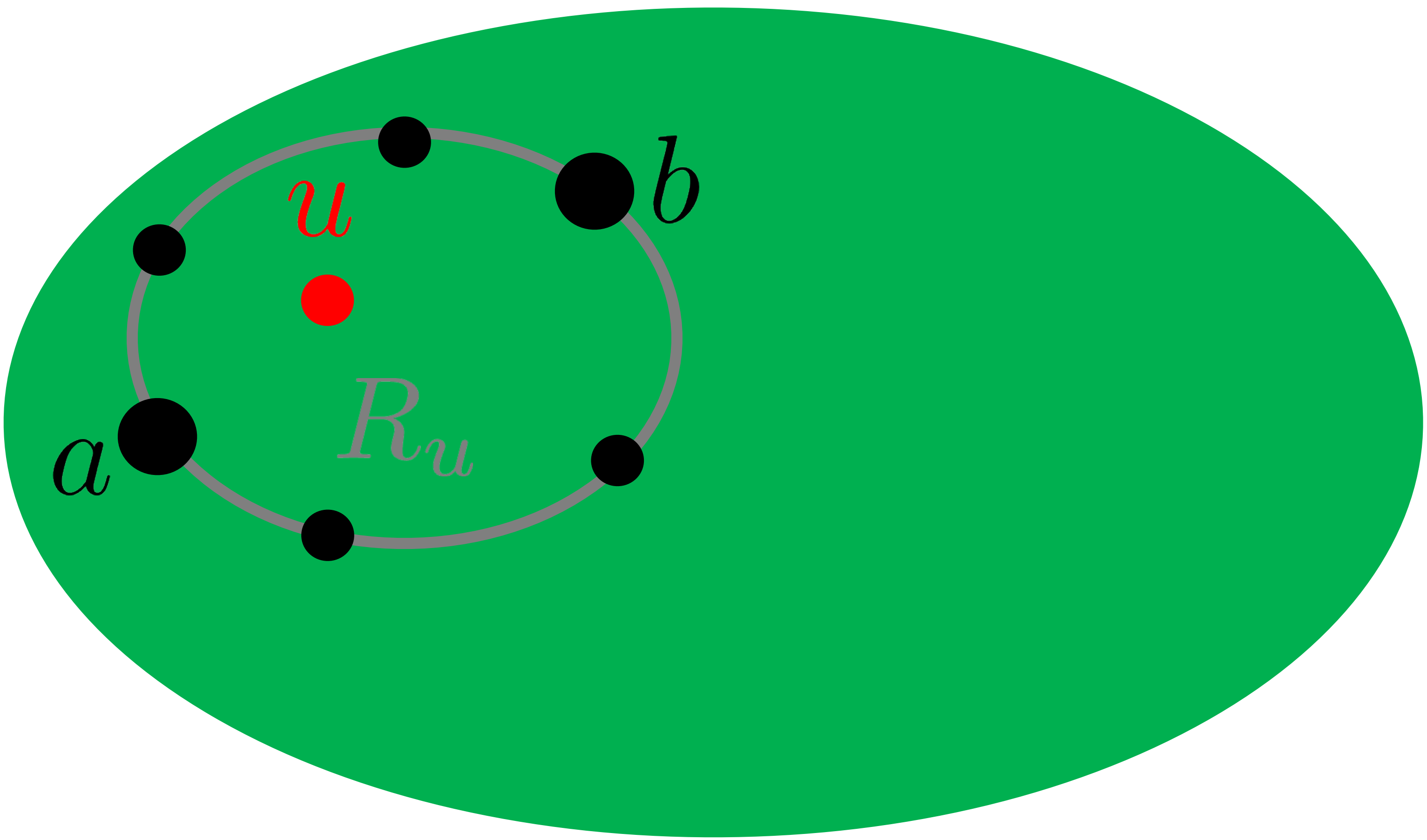
∂R_u to ∂R_u in $G \backslash \{u\}$



u's label:

u to ∂R internally disjoint from R for every region R

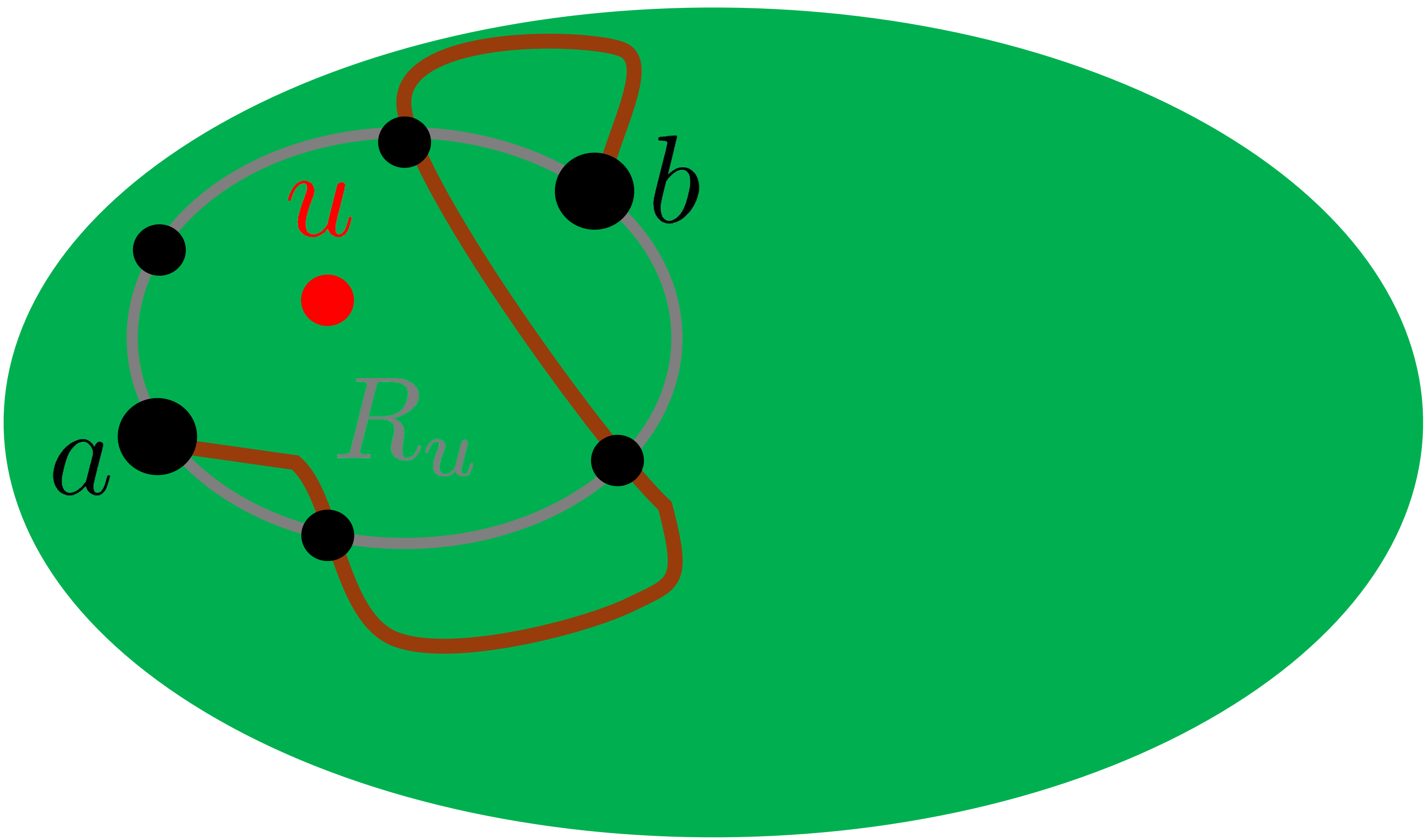
∂R_u to ∂R_u in $G \backslash \{u\}$



u's label:

u to ∂R internally disjoint from R for every region R

∂R_u to ∂R_u in $G \backslash \{u\}$

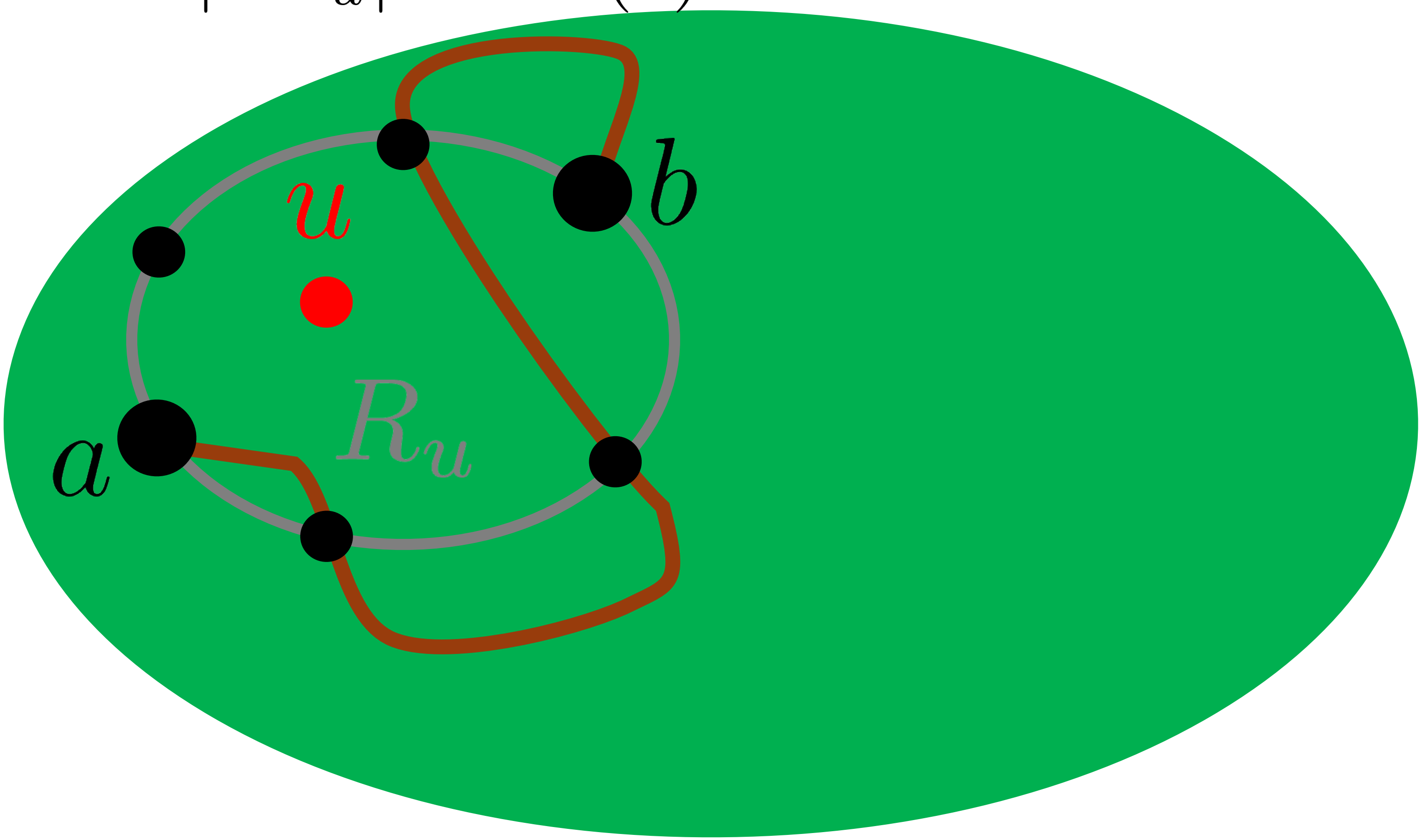


u's label:

u to ∂R internally disjoint from R for every region R

∂R_u to ∂R_u in $G \backslash \{u\}$

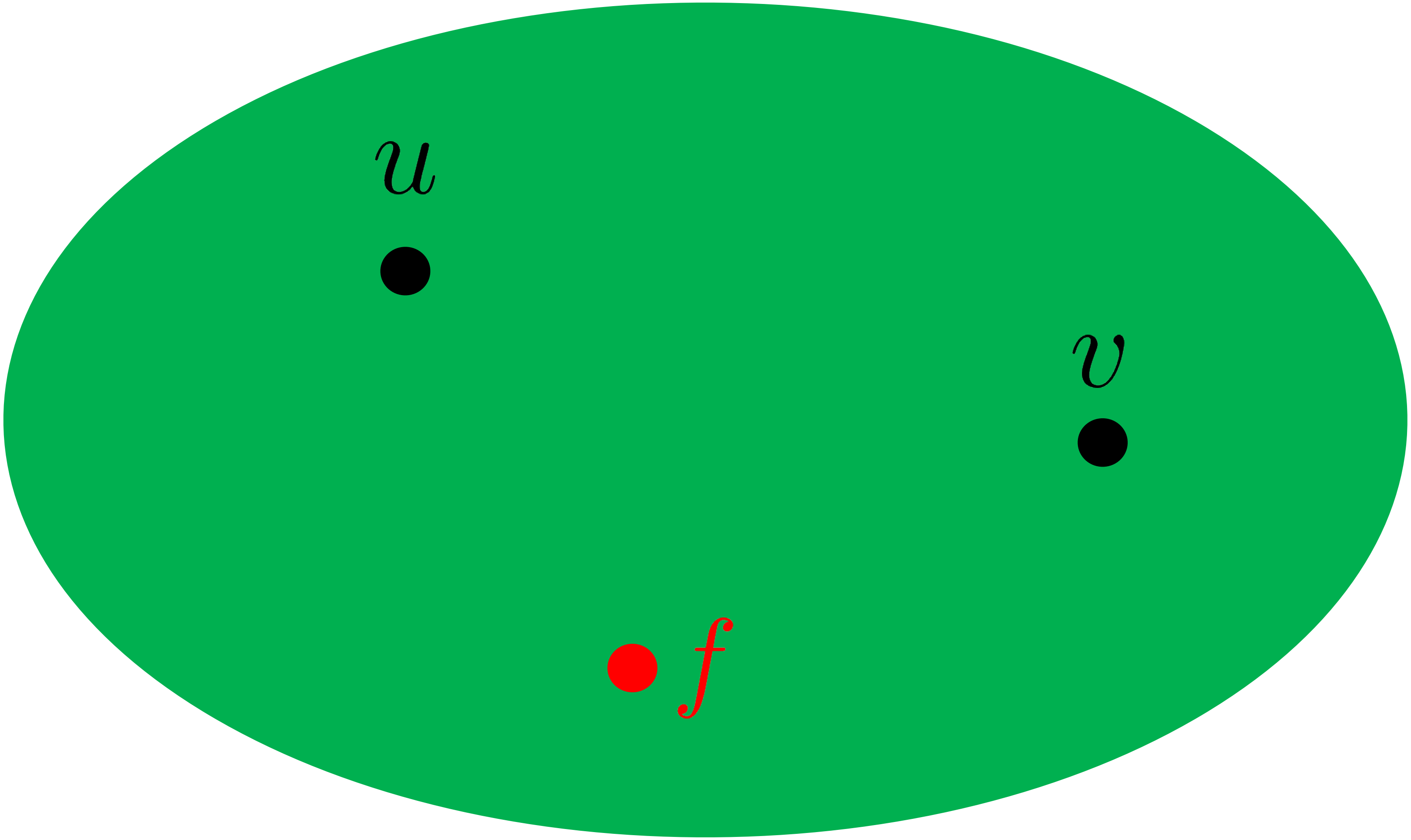
$\text{space} = |\partial R_u|^2 = \tilde{O}(r)$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

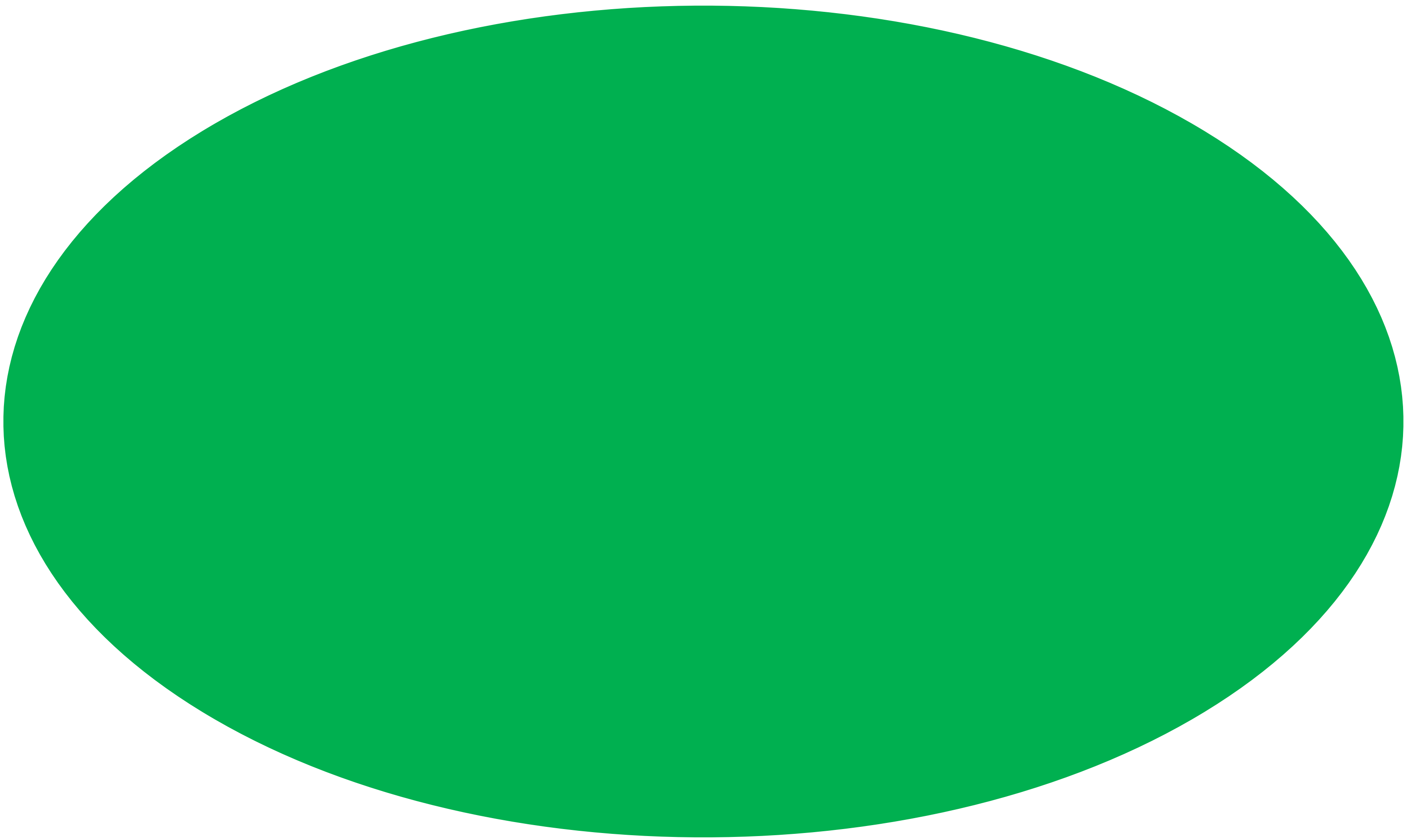
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

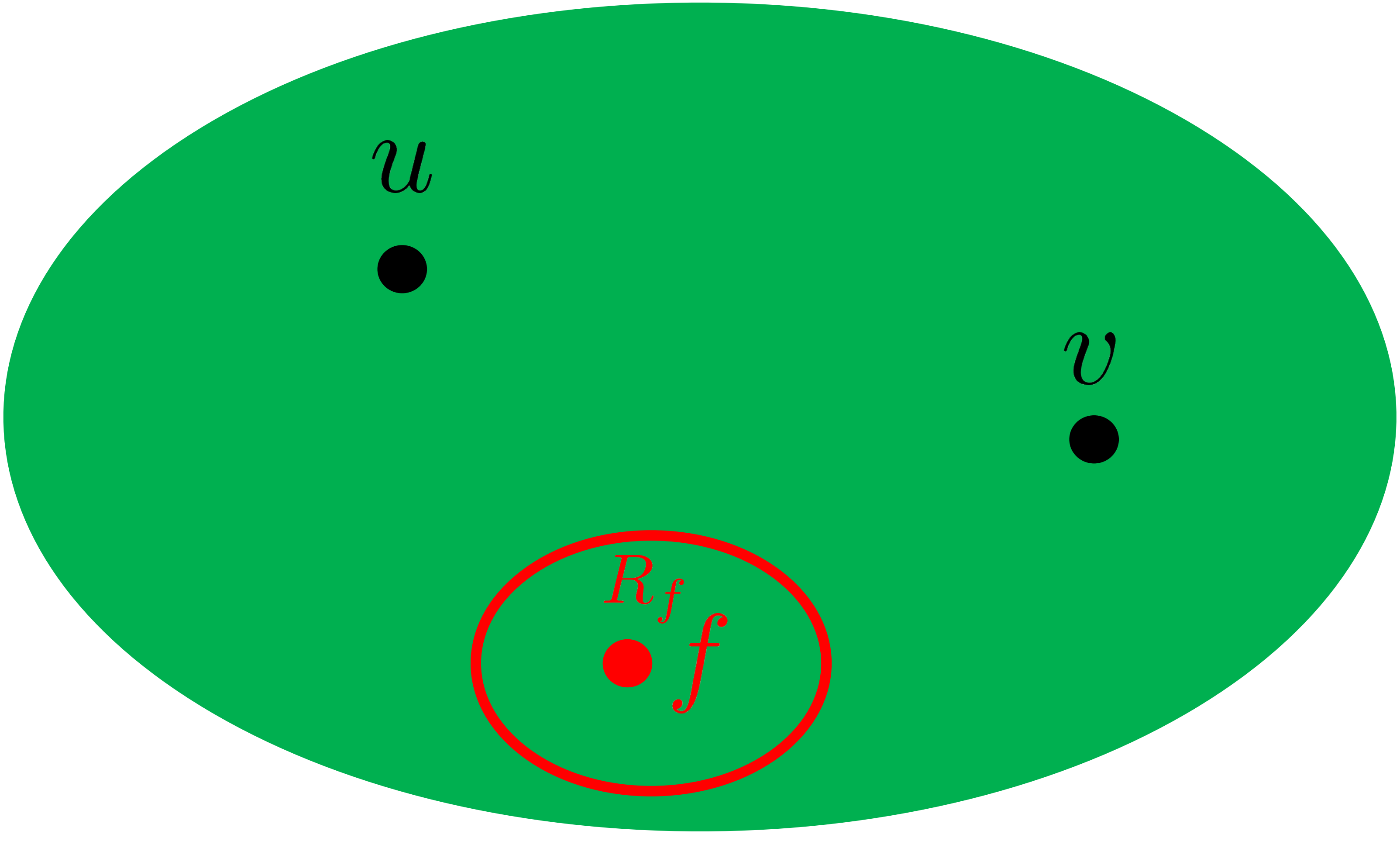
∂R_u to ∂R_u in $G\backslash\{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

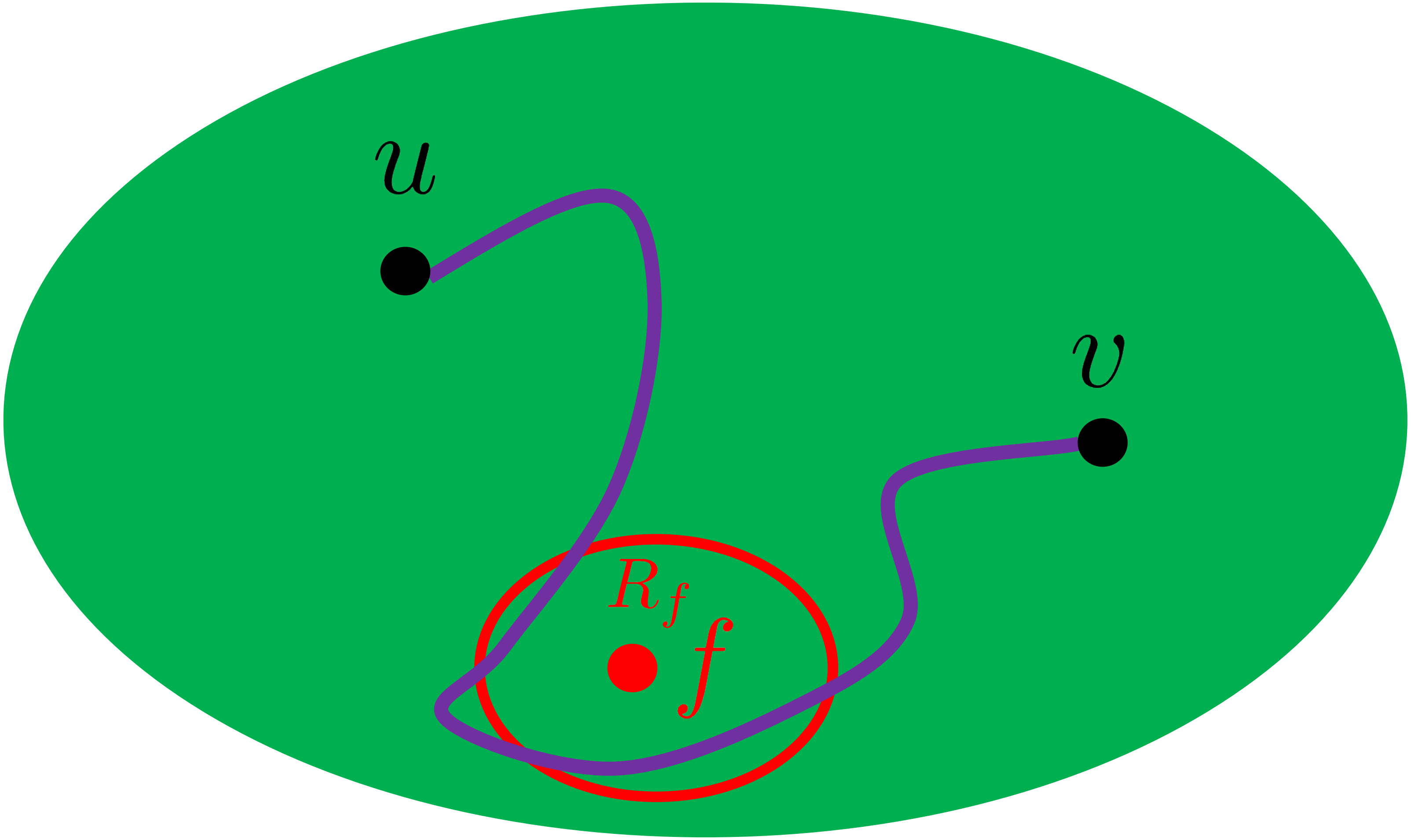
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

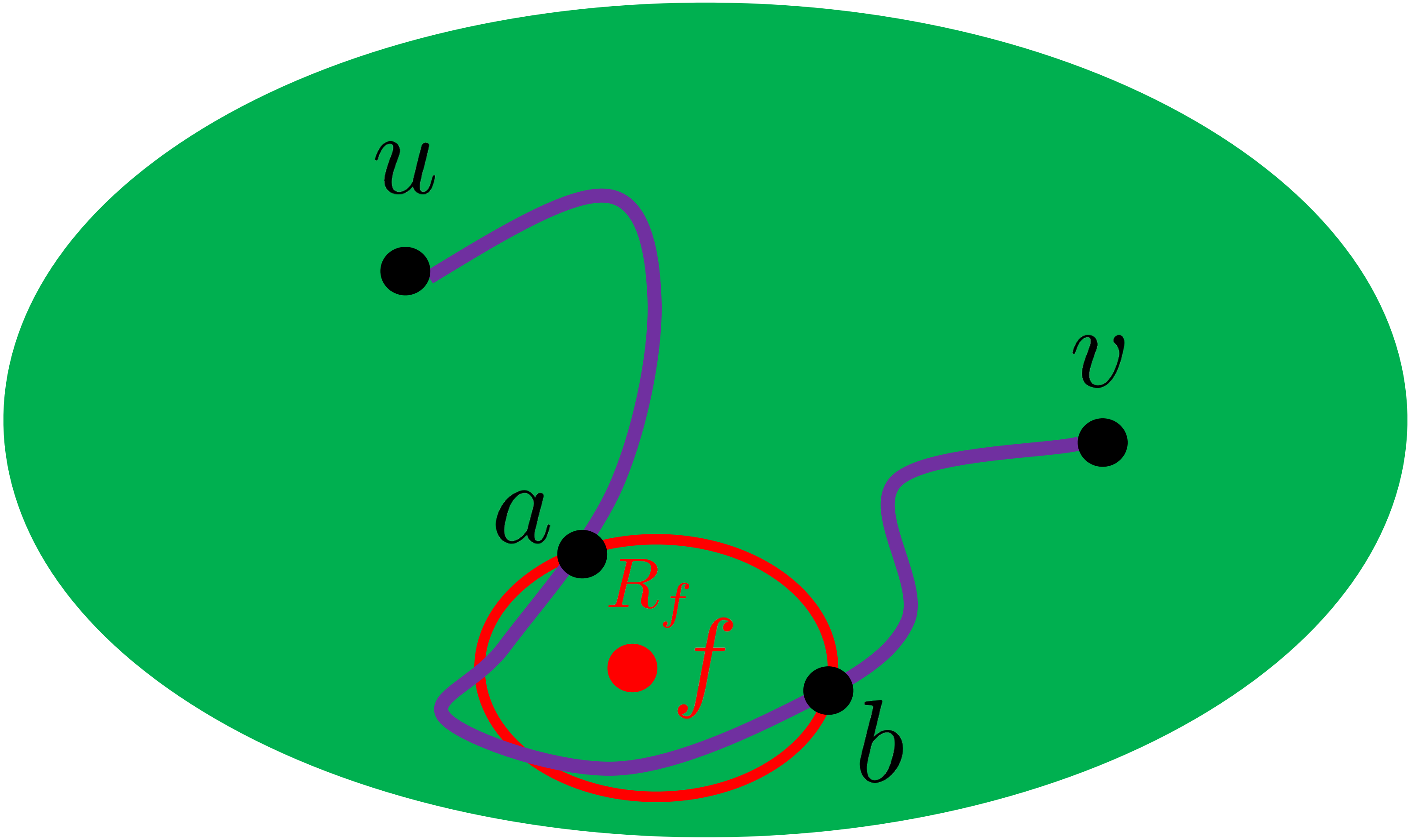
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

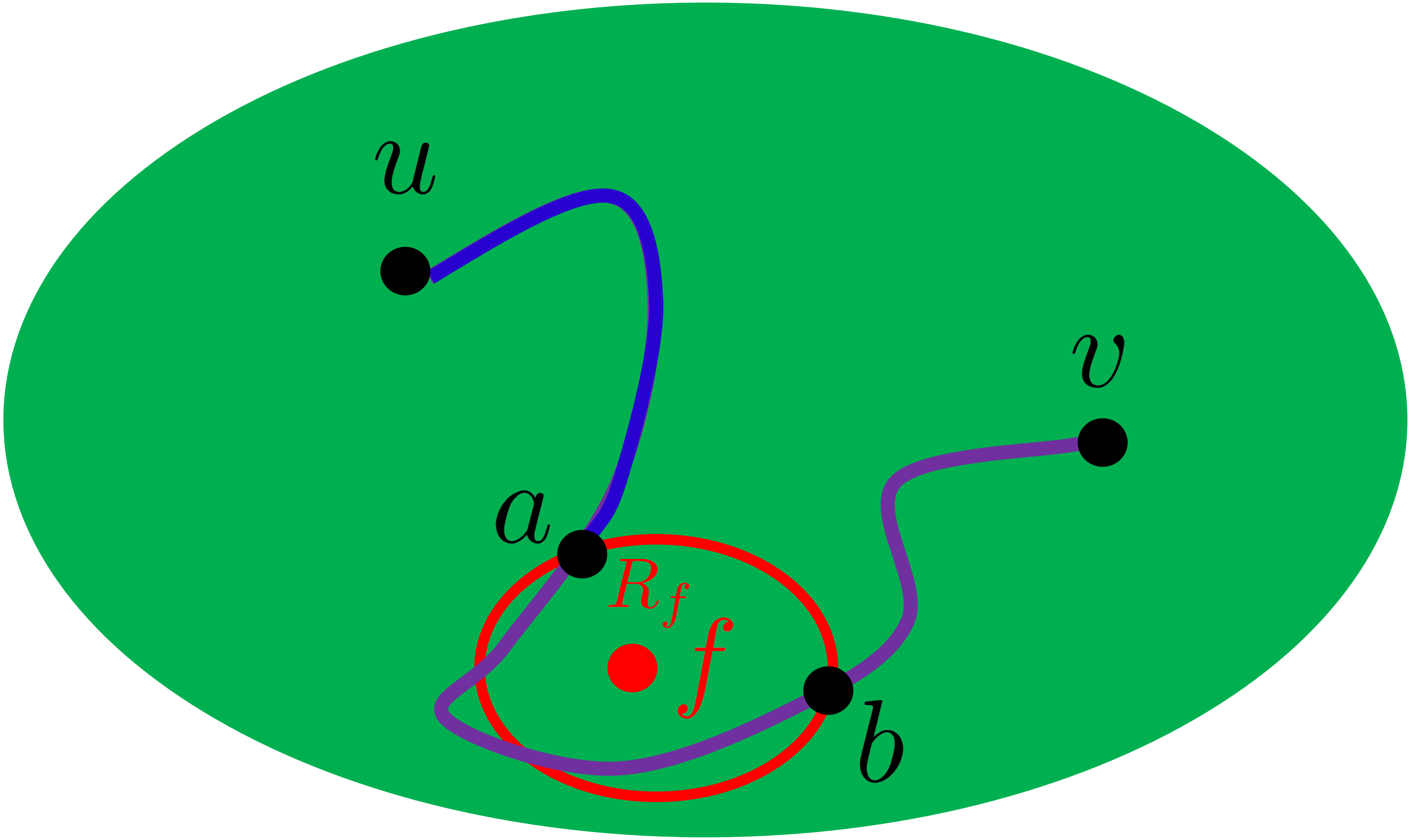
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

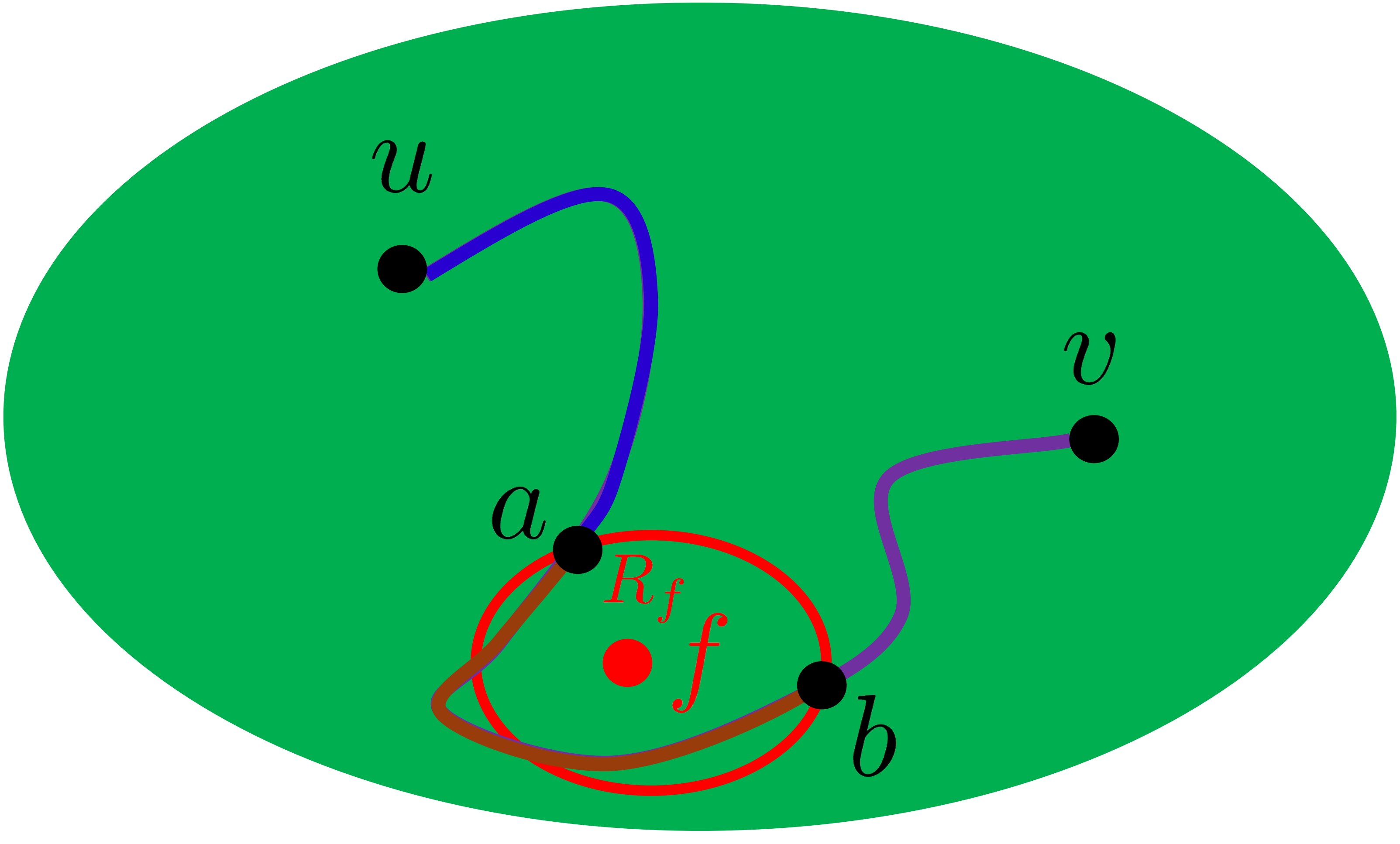
∂R_u to ∂R_u in $G\backslash\{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

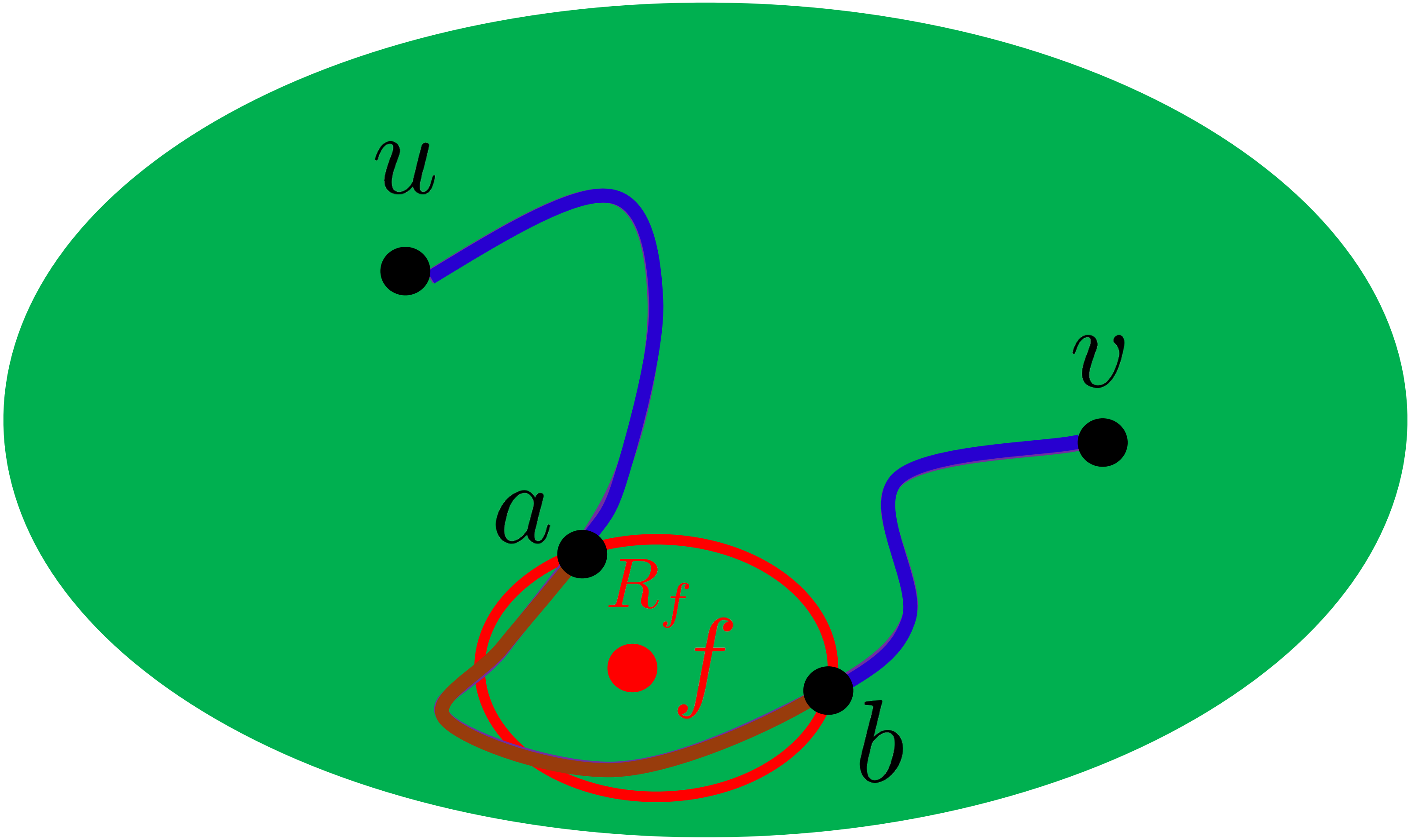
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

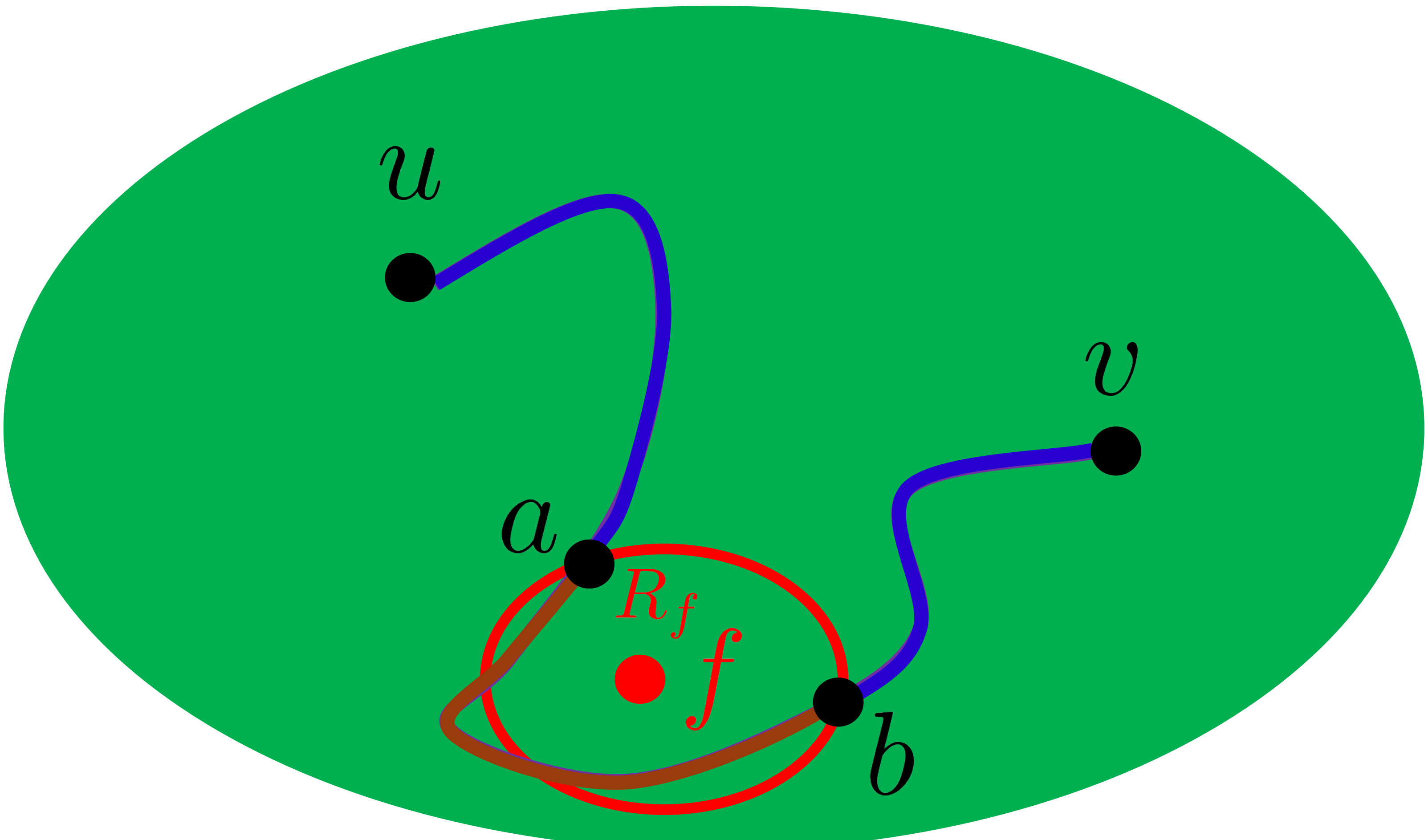
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

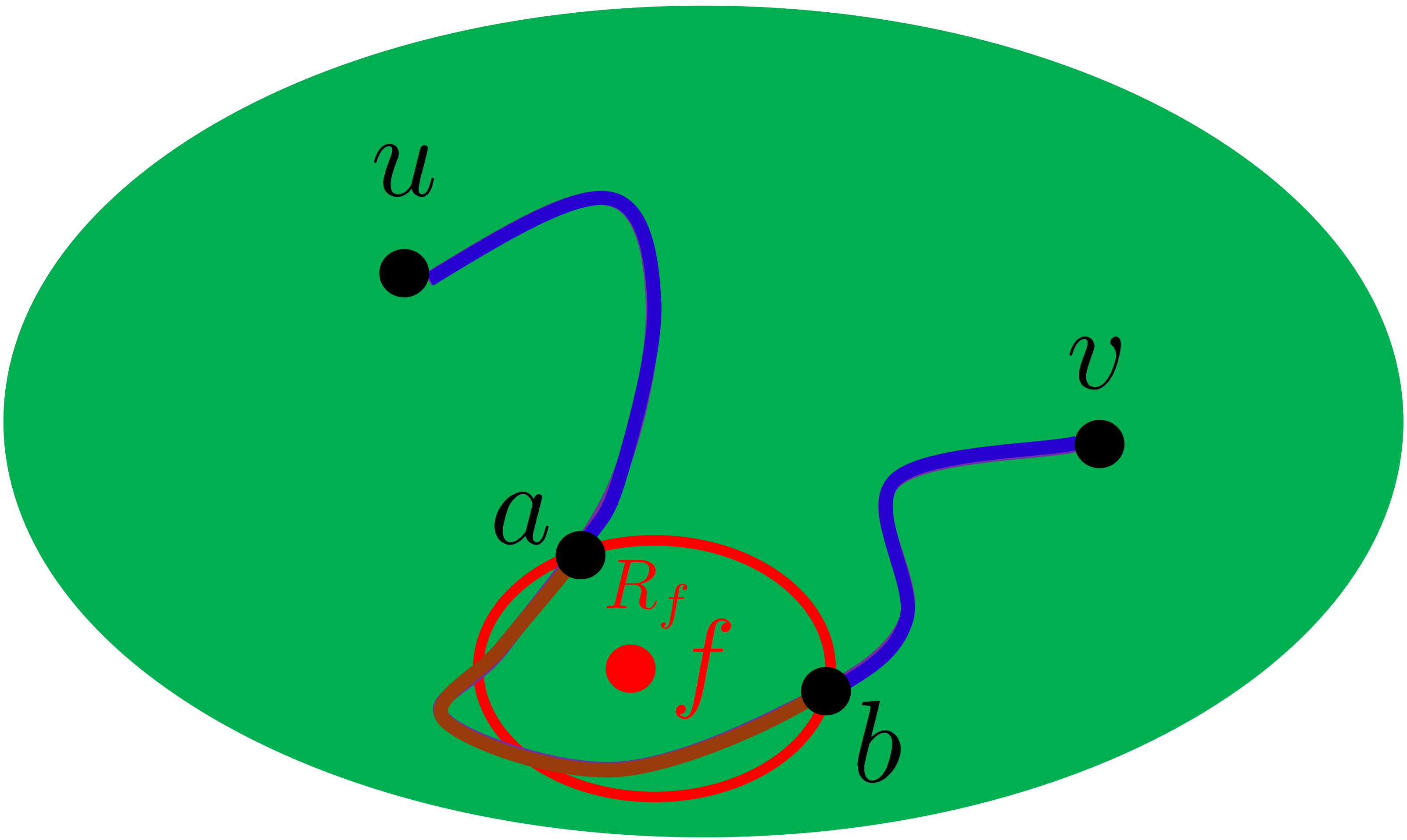
∂R_u to ∂R_u in $G \backslash \{u\}$



Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

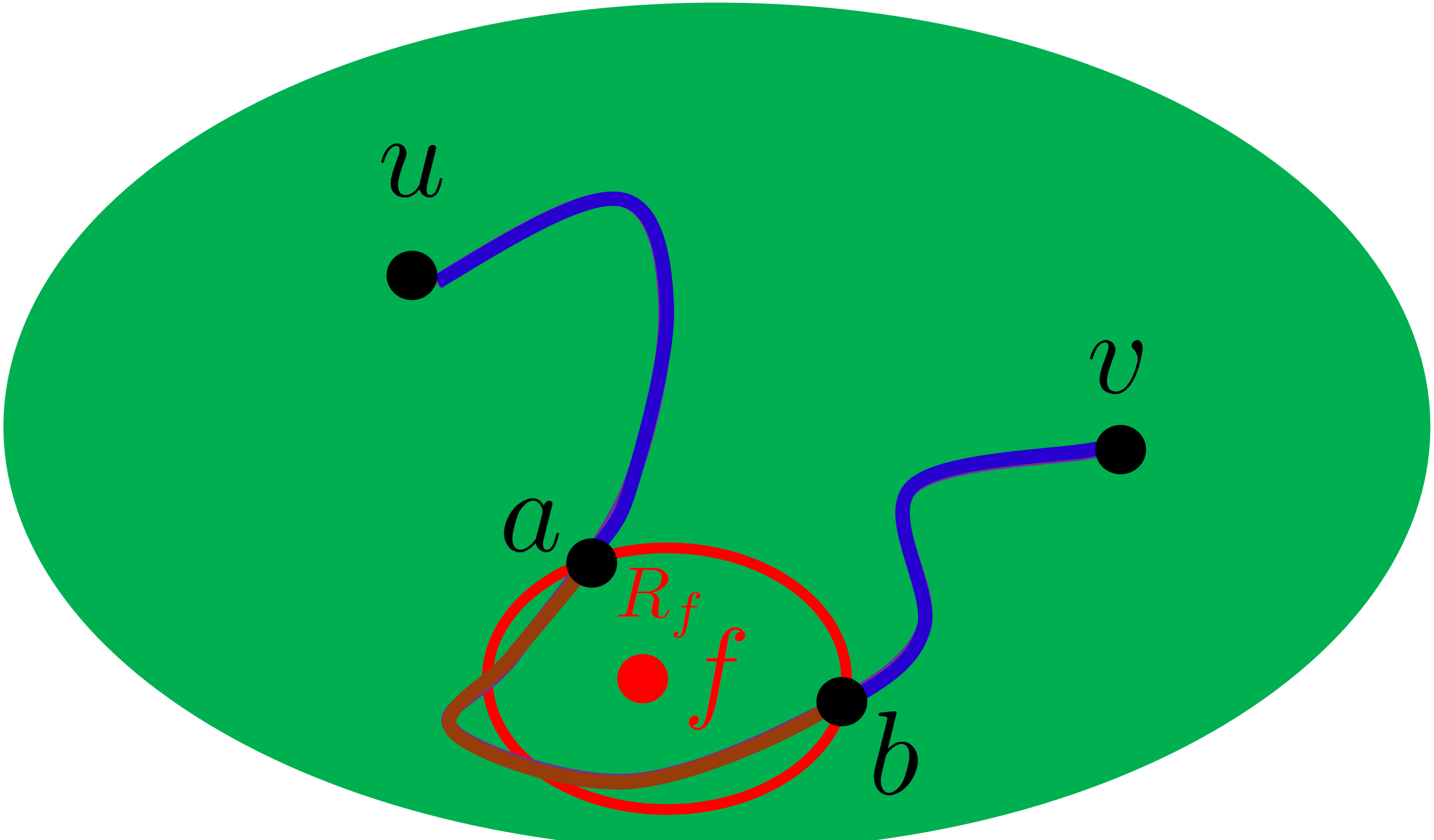
∂R_u to ∂R_u in $G \backslash \{u\}$



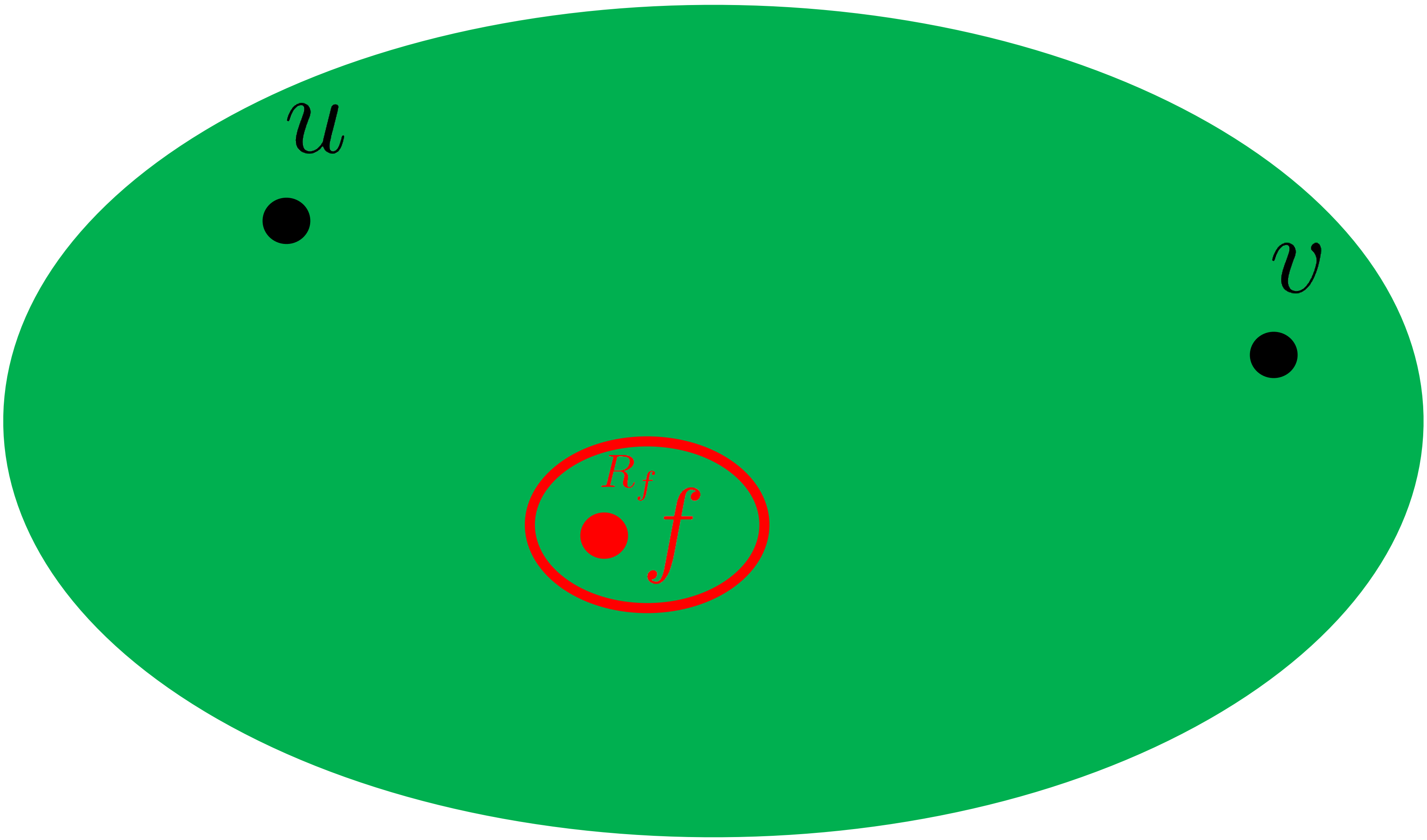
Case 1: Shortest path touches ∂R_f

u to ∂R internally disjoint from R for every region R

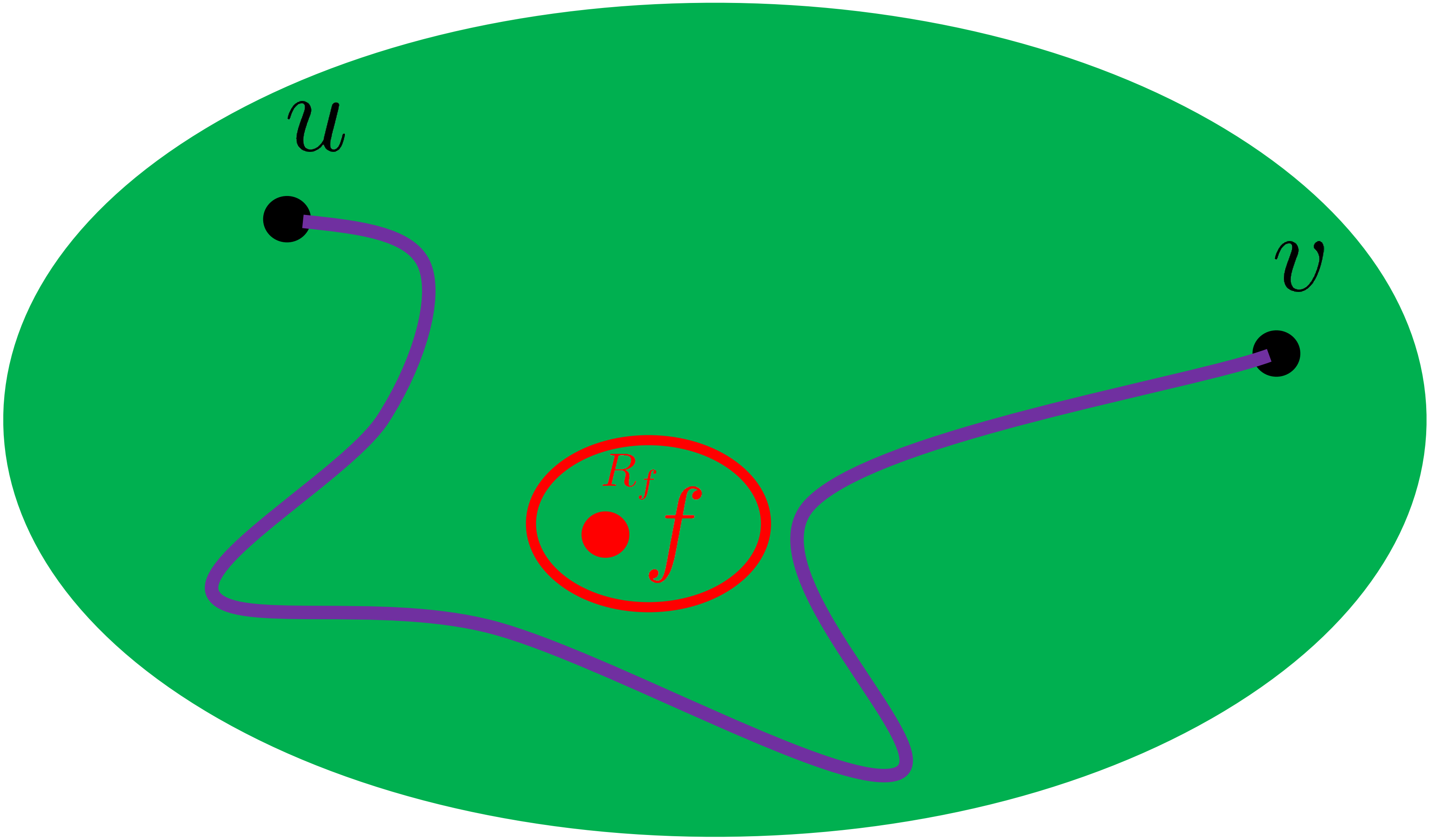
∂R_u to ∂R_u in $G \backslash \{u\}$



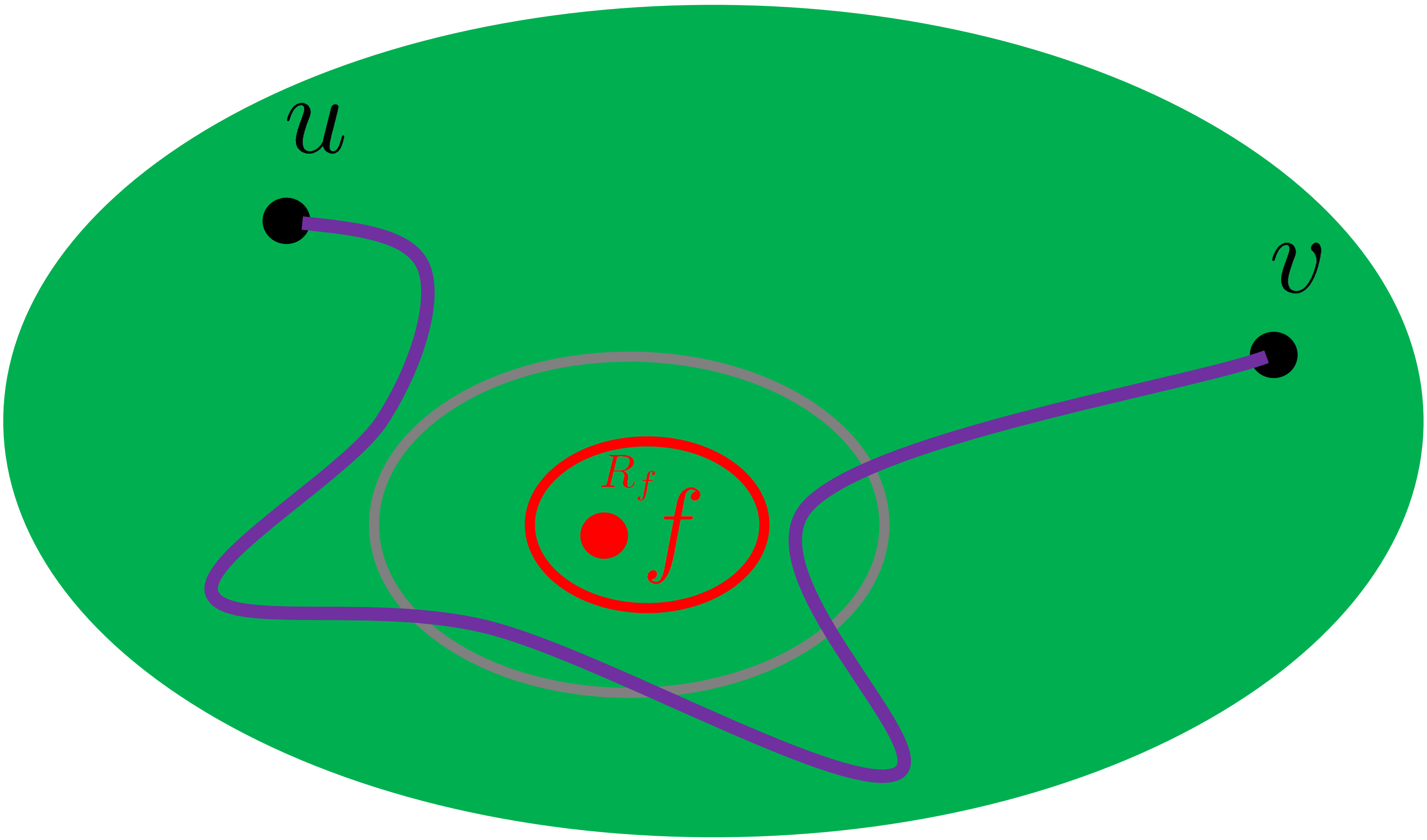
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



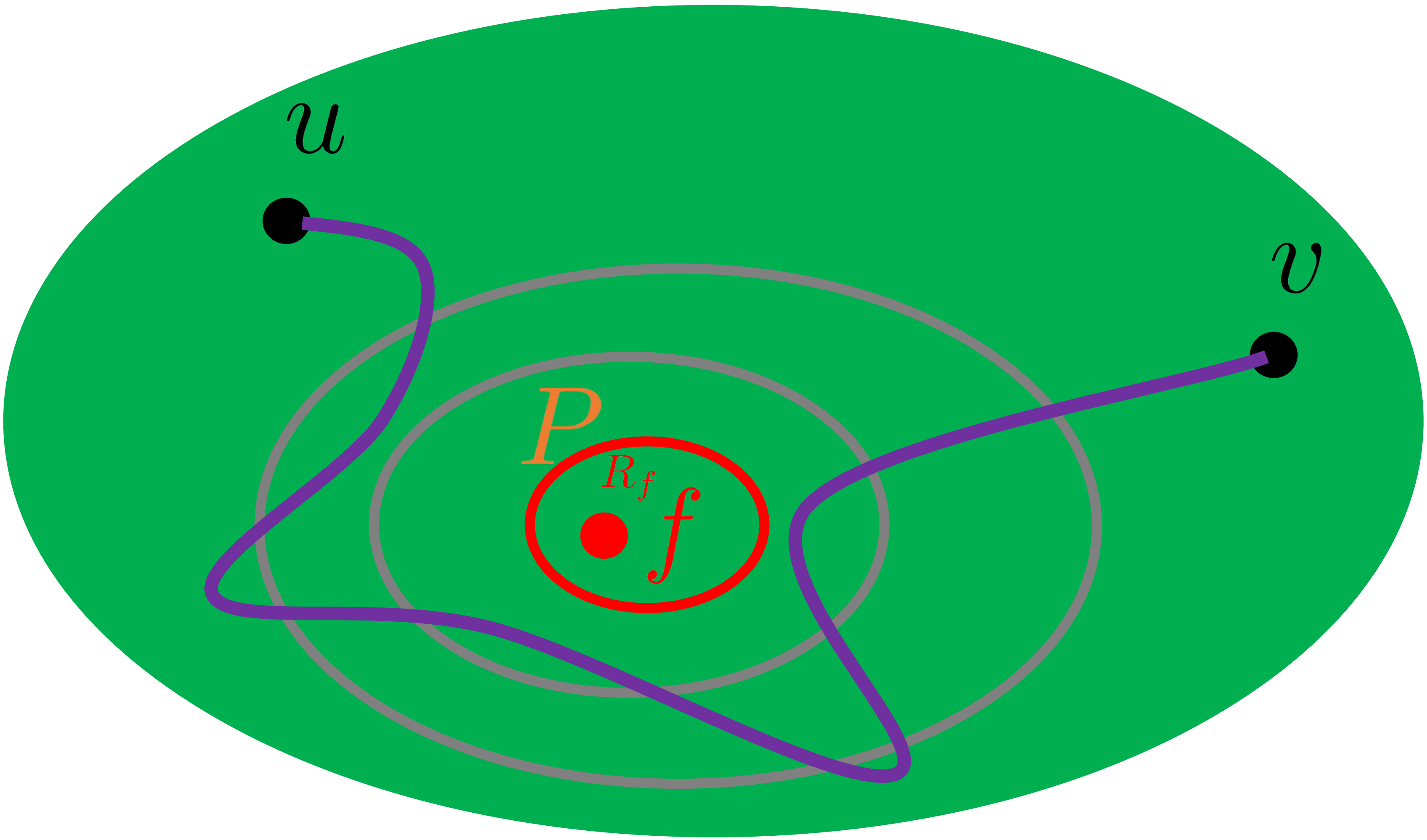
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



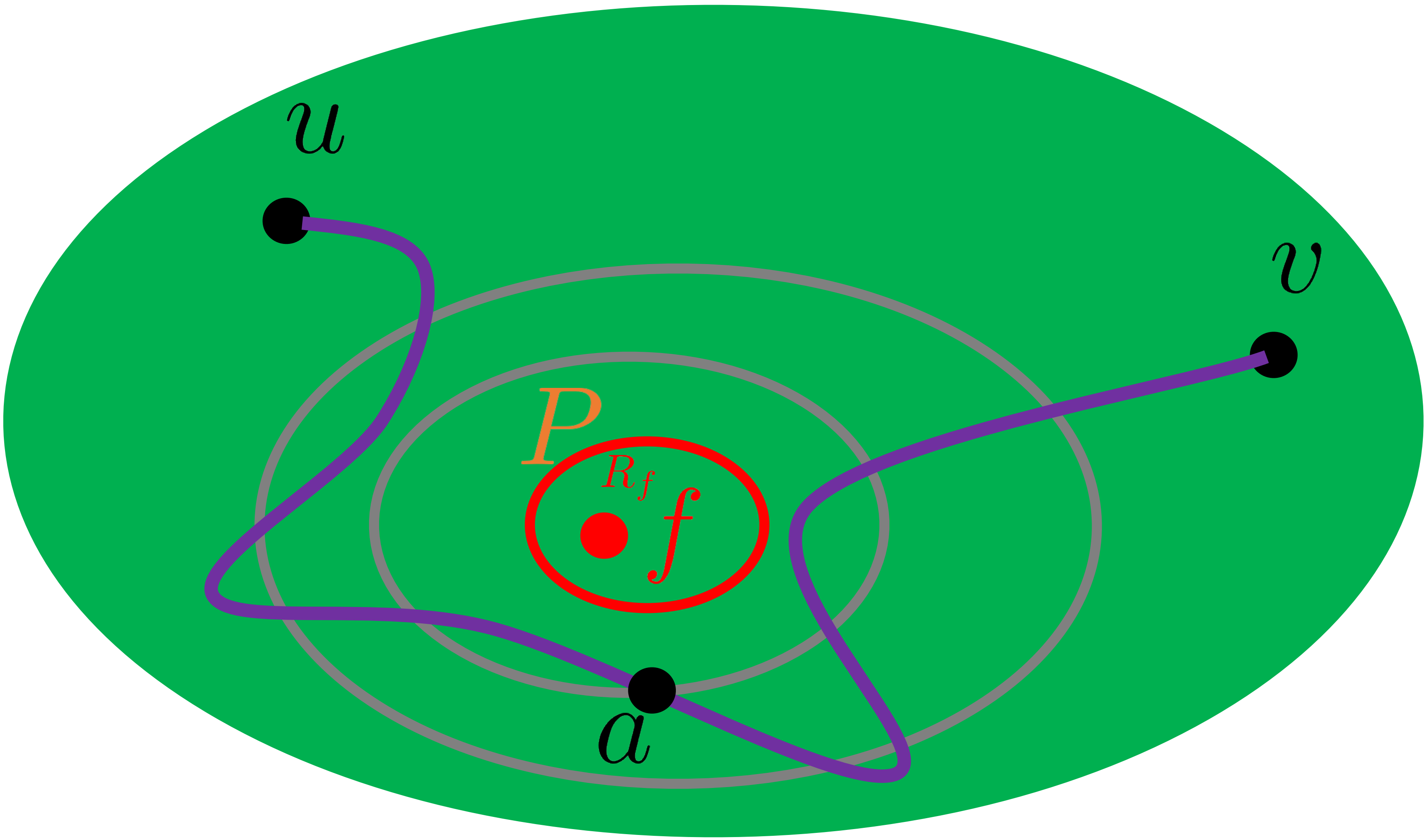
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



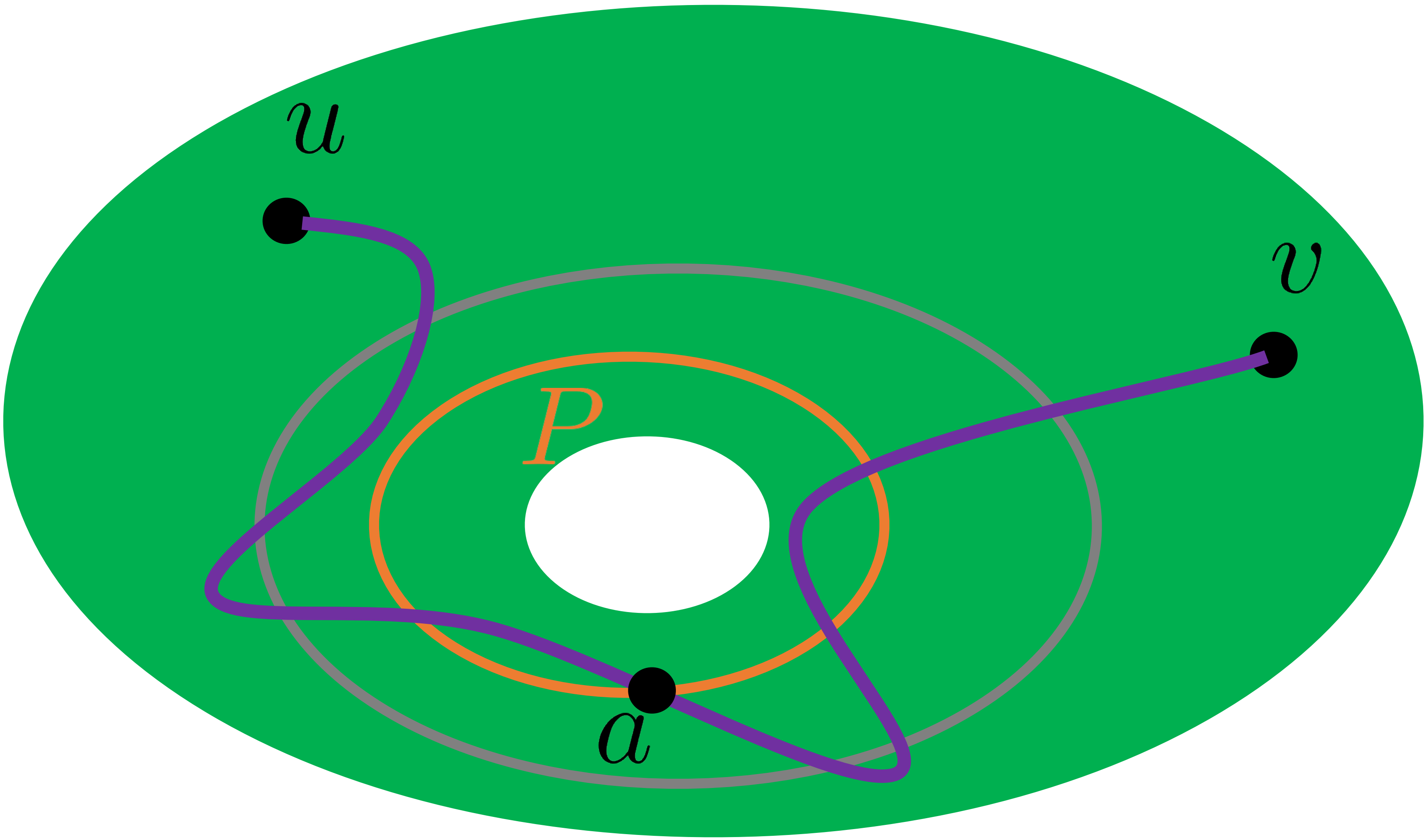
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



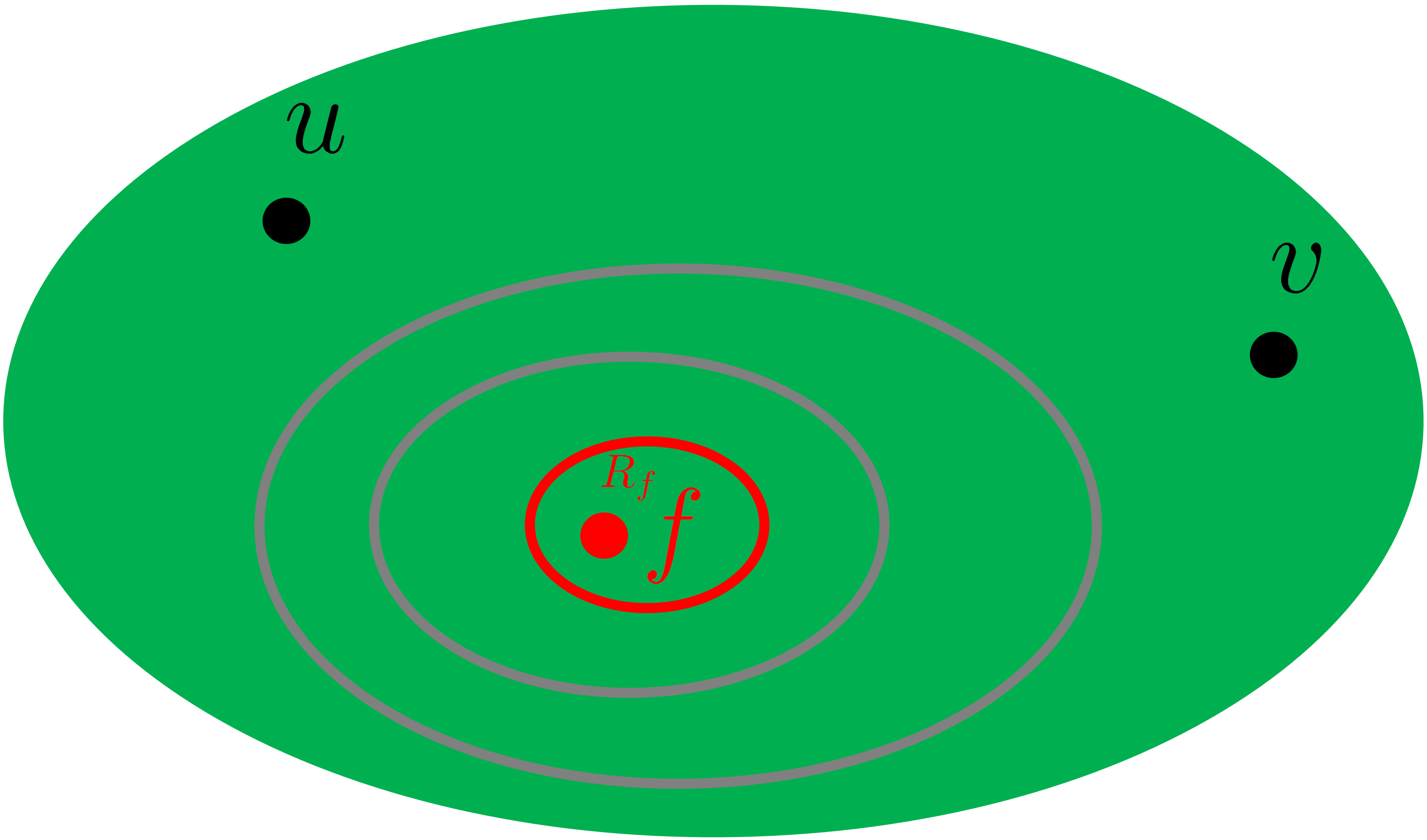
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



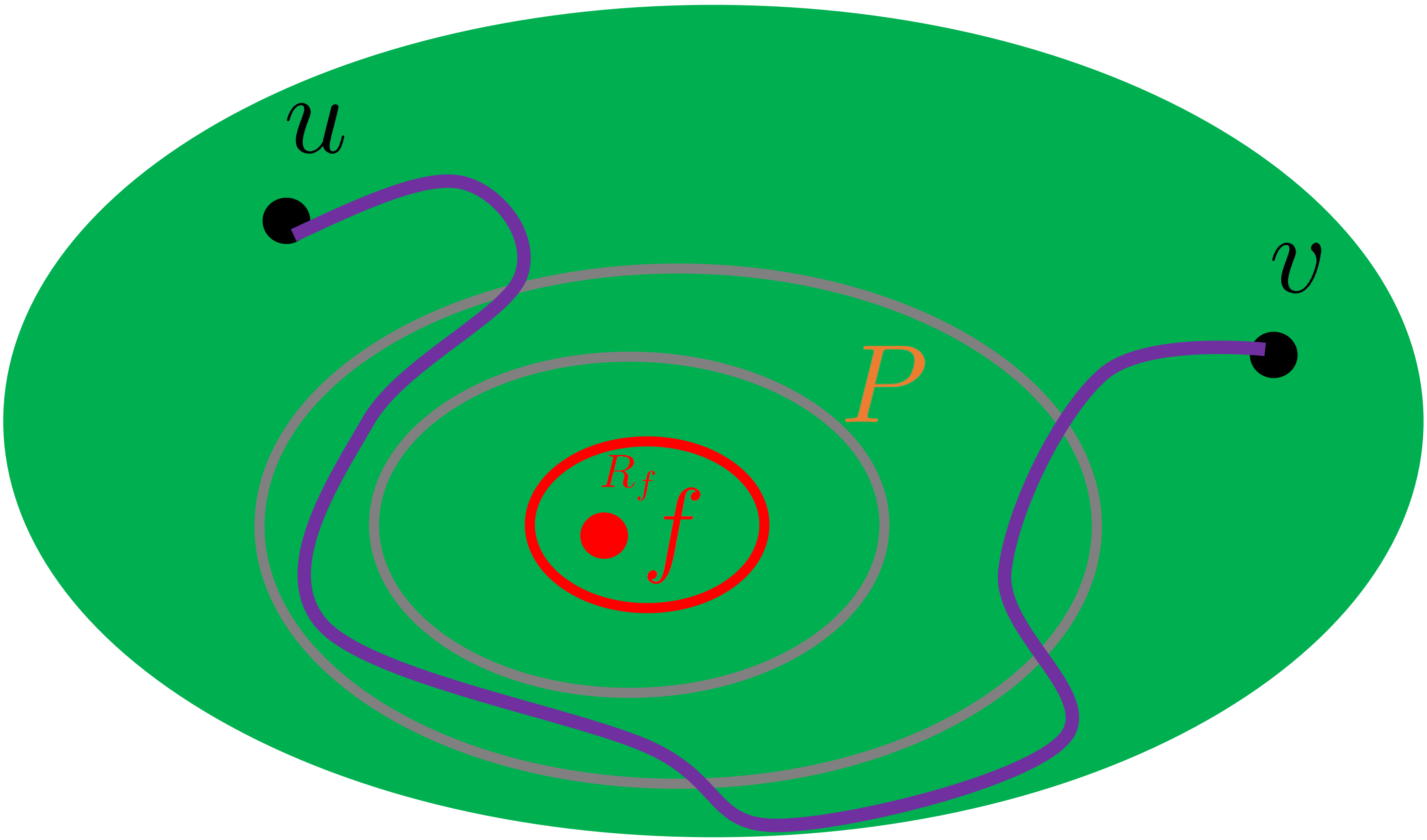
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



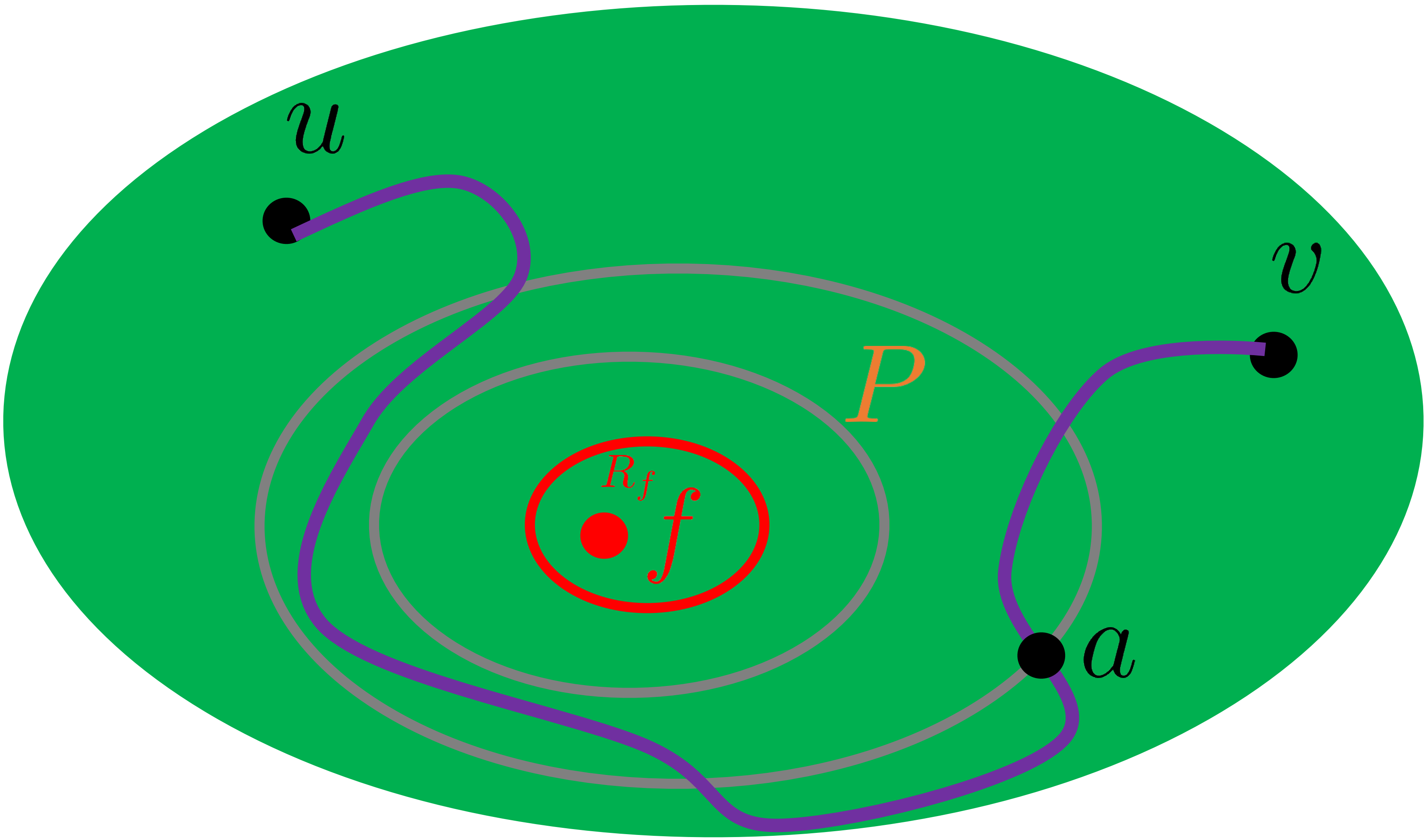
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



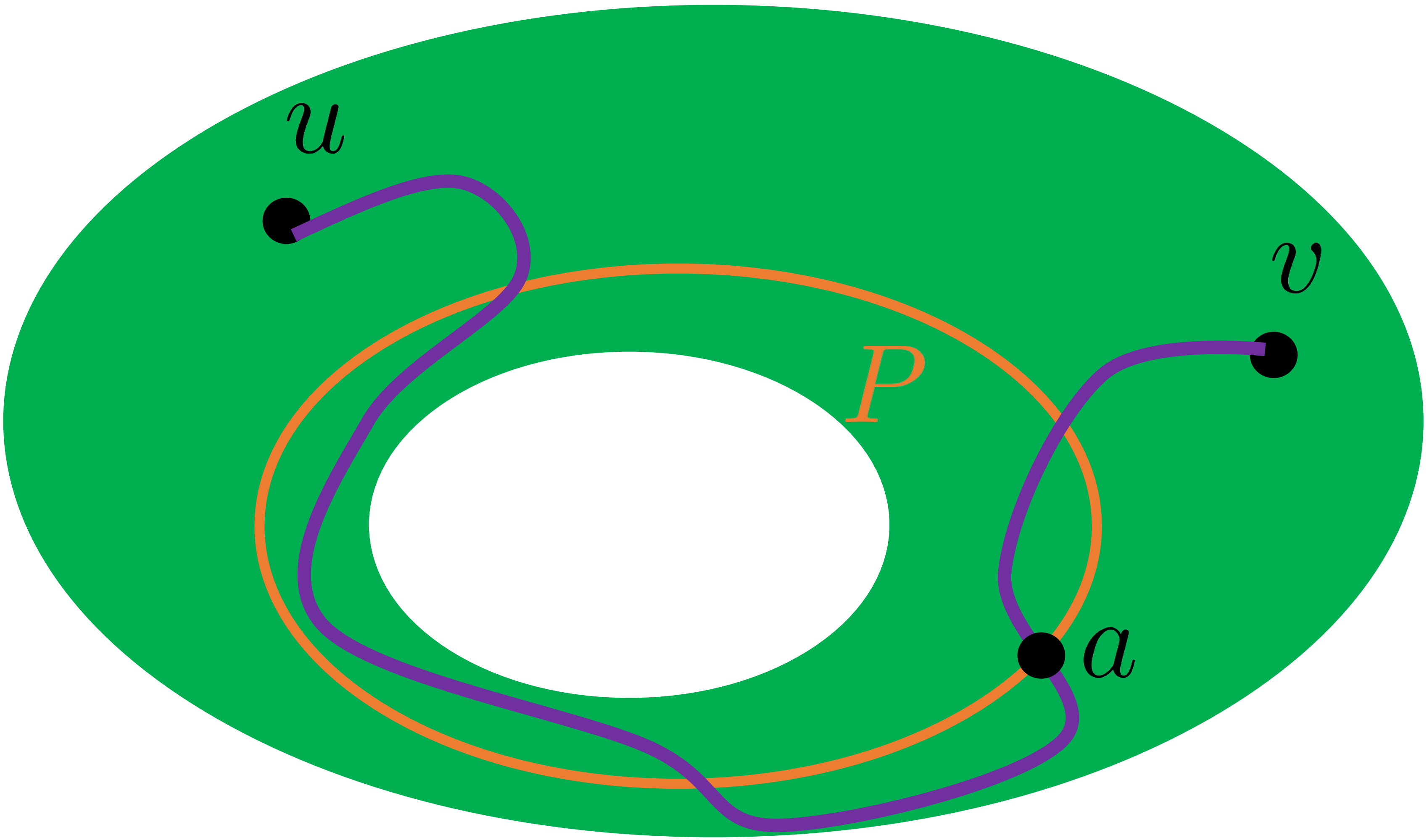
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



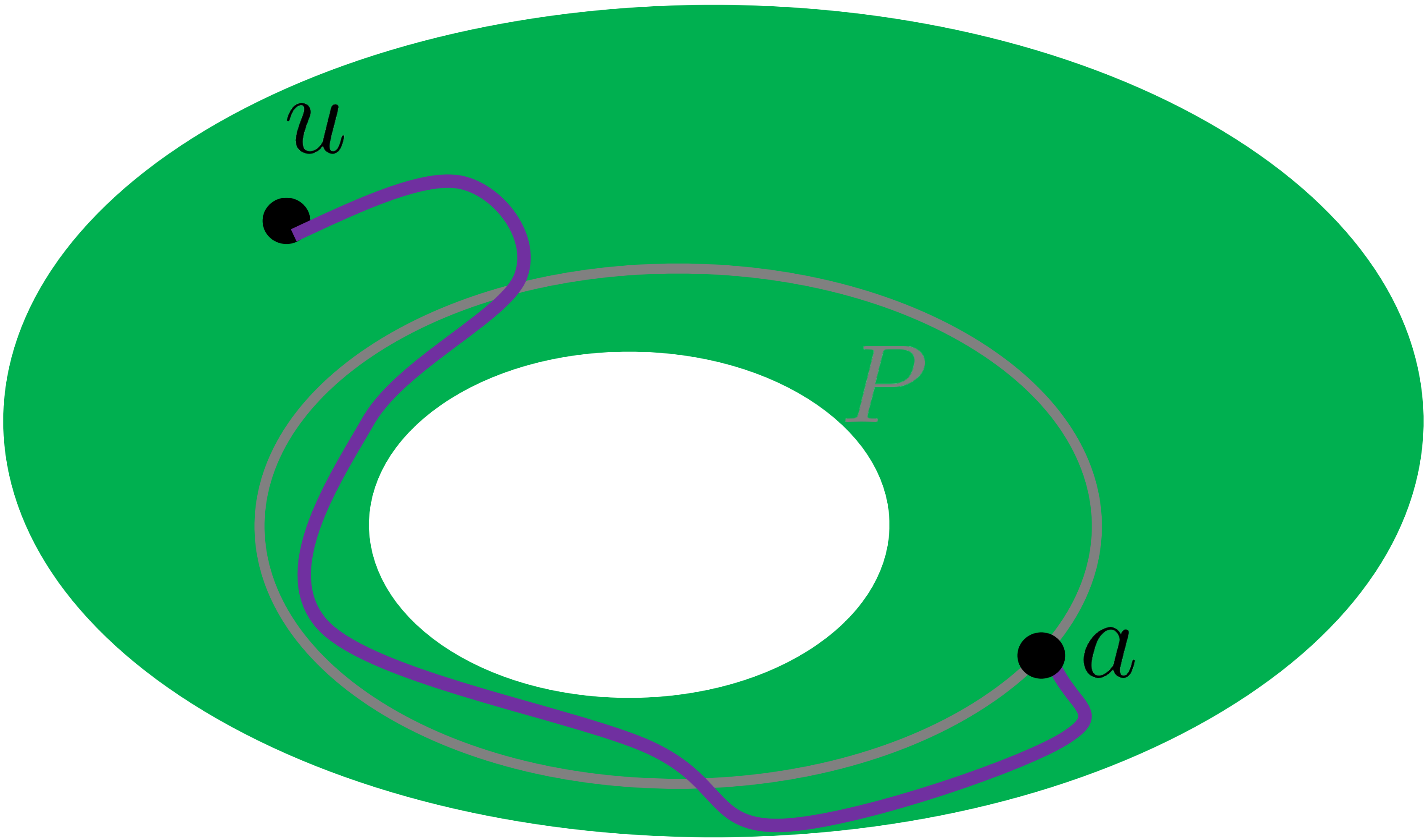
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



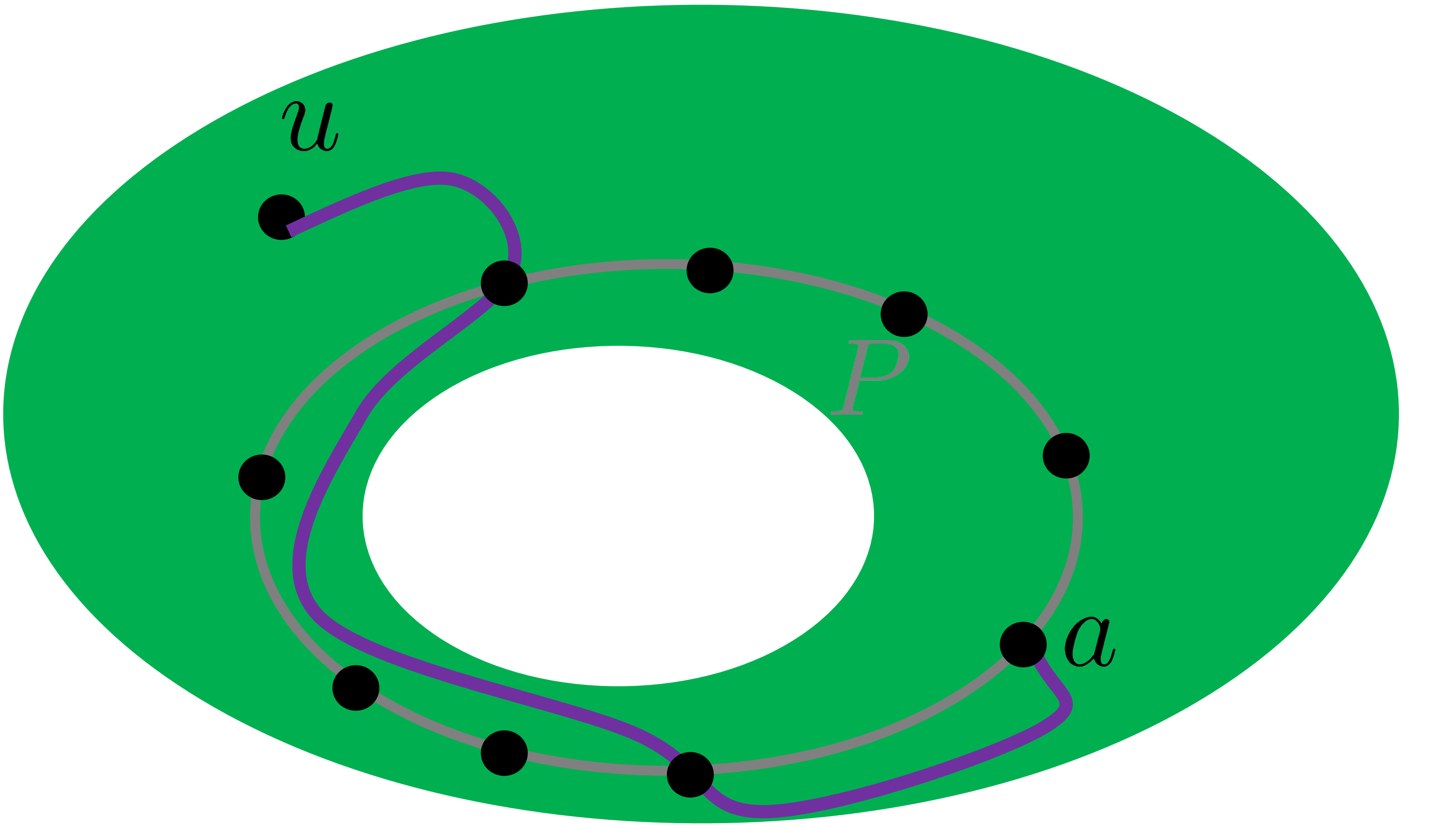
Case 2: Shortest path touches ∂P for ancestor piece P of R_f



u's label in case 2:

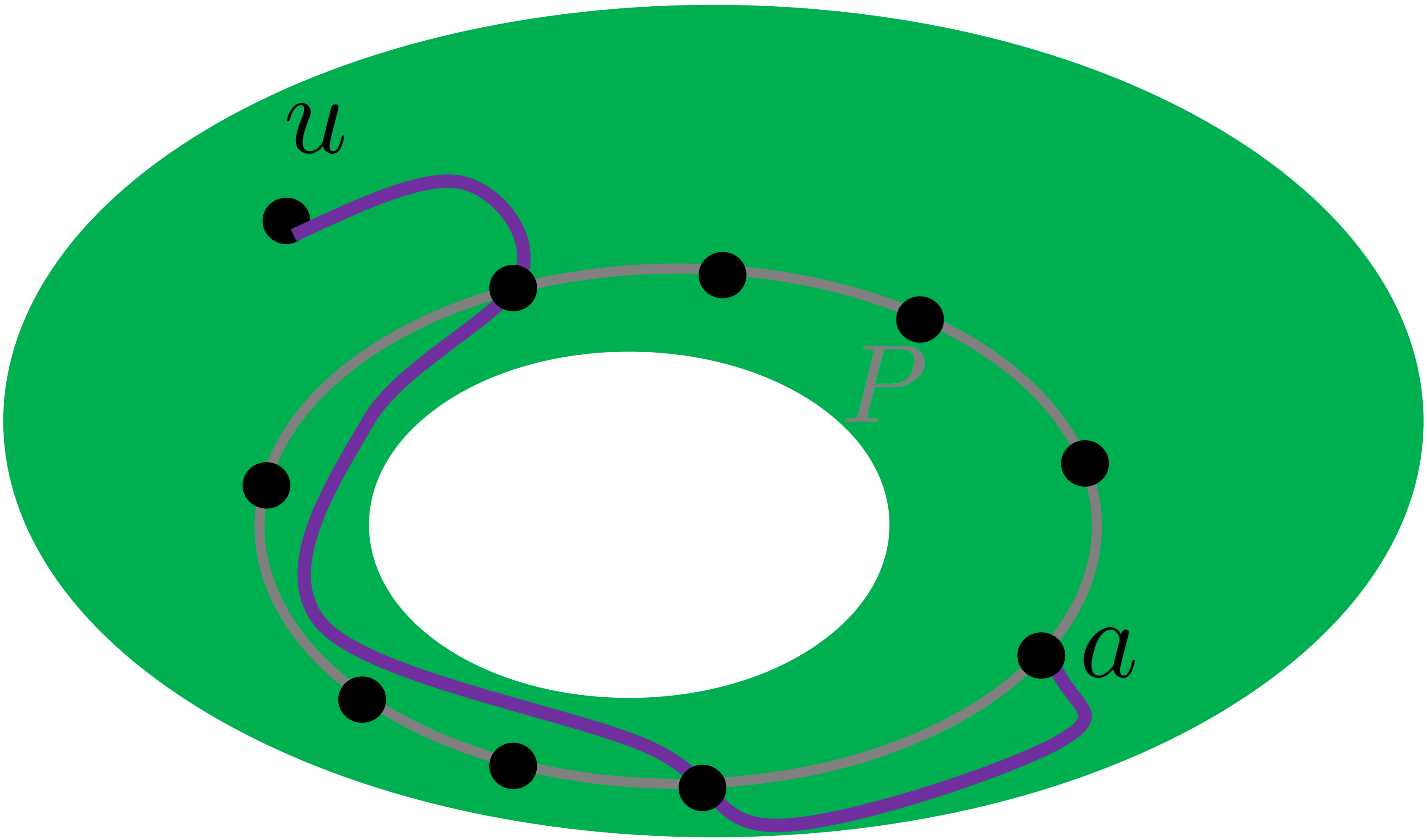


u's label in case 2:

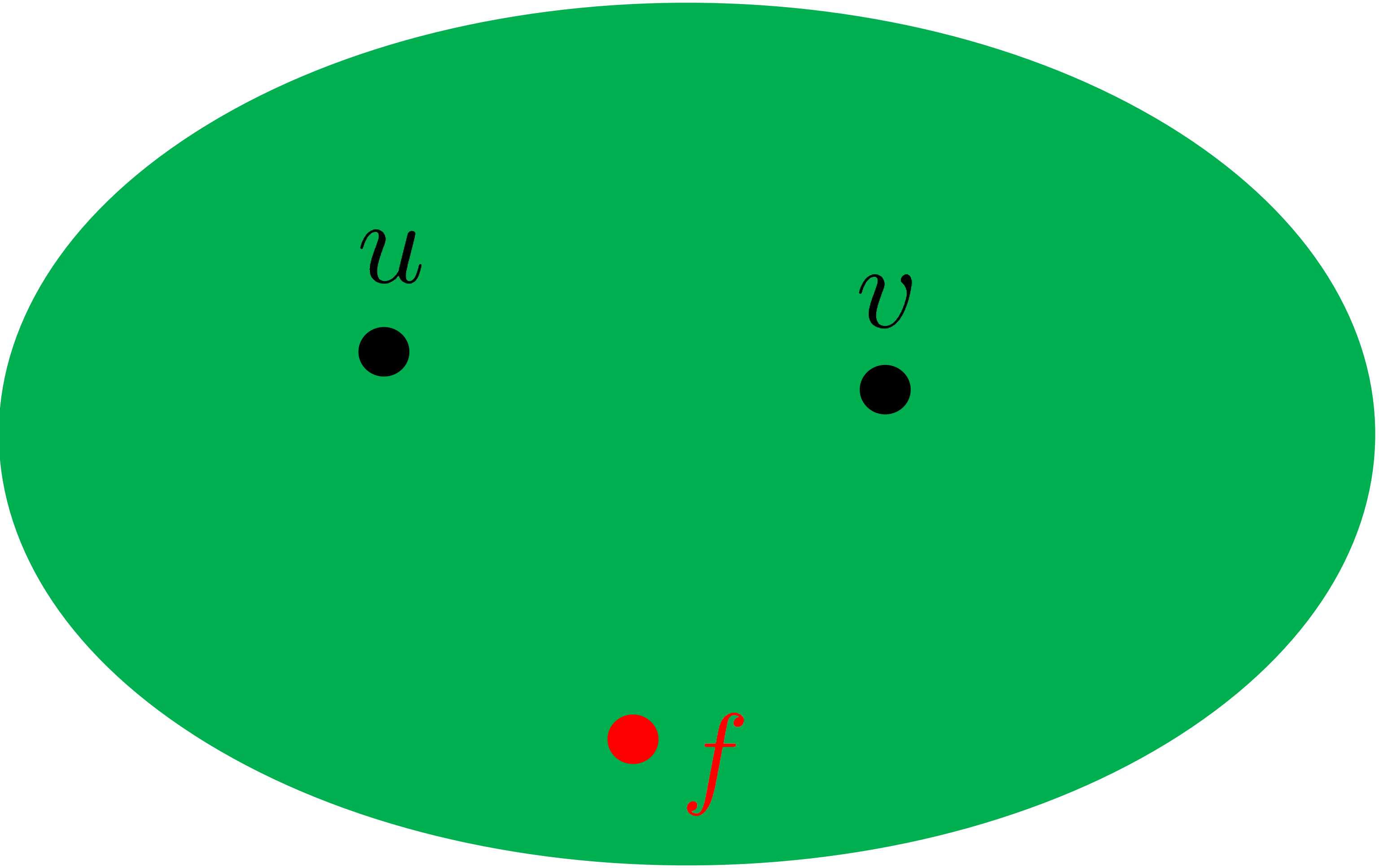


u's label in case 2:

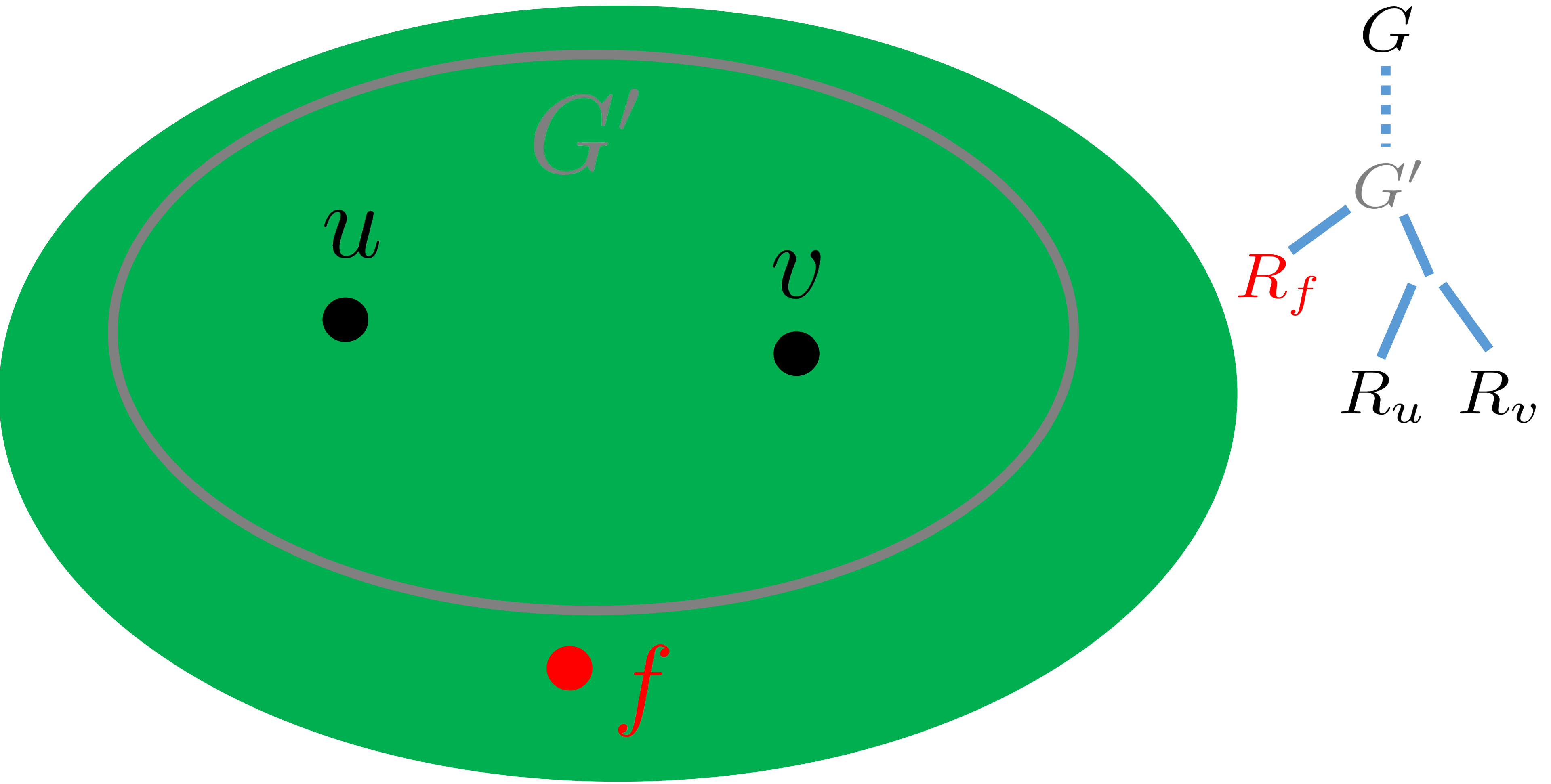
$$\text{space} = \tilde{O}(\sum_{P \in Pieces} |\partial P|) = \tilde{O}(n/\sqrt{r})$$



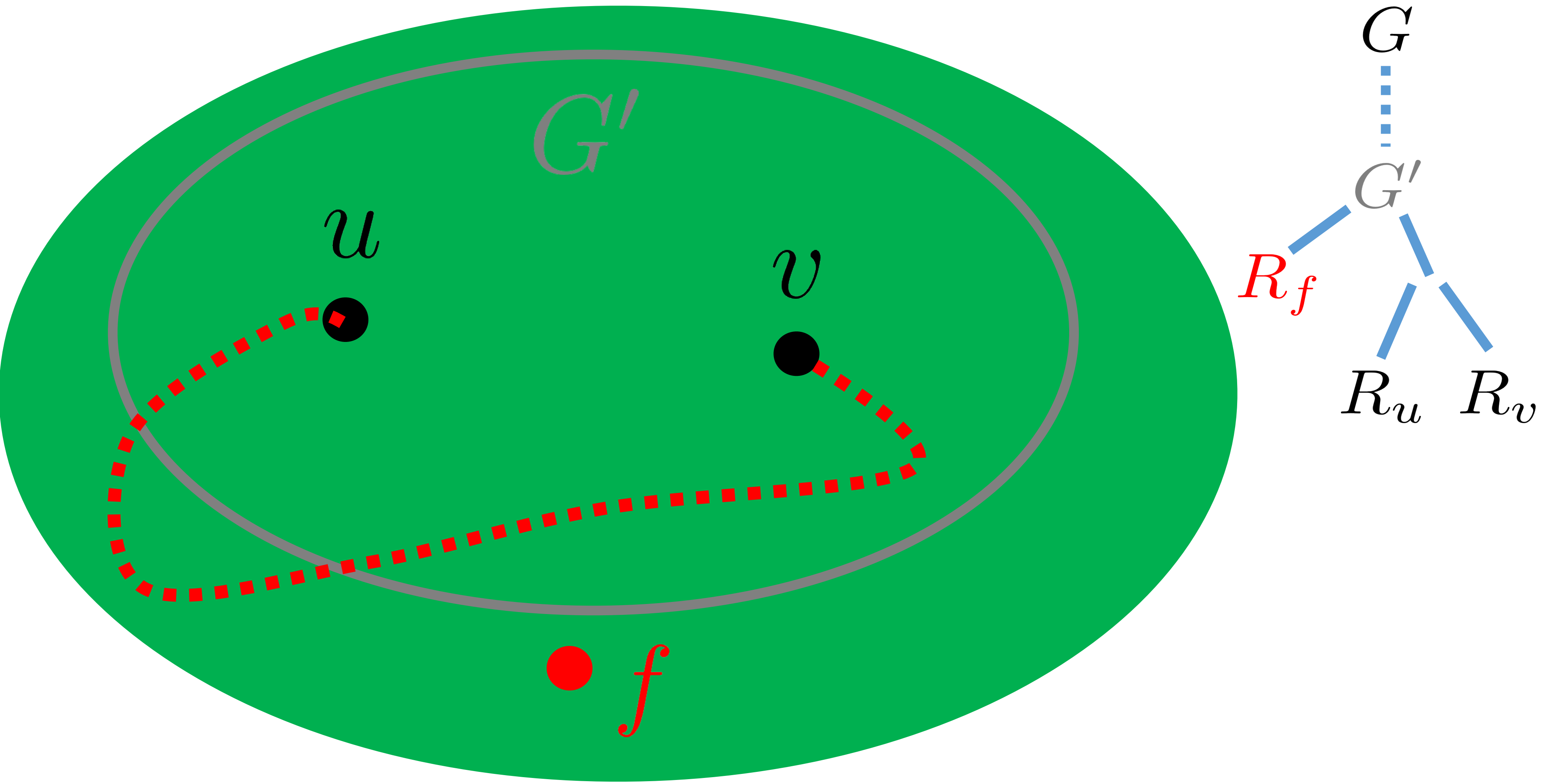
Case 3: Shortest path doesn't touch ∂P for any ancestor piece P of R_f



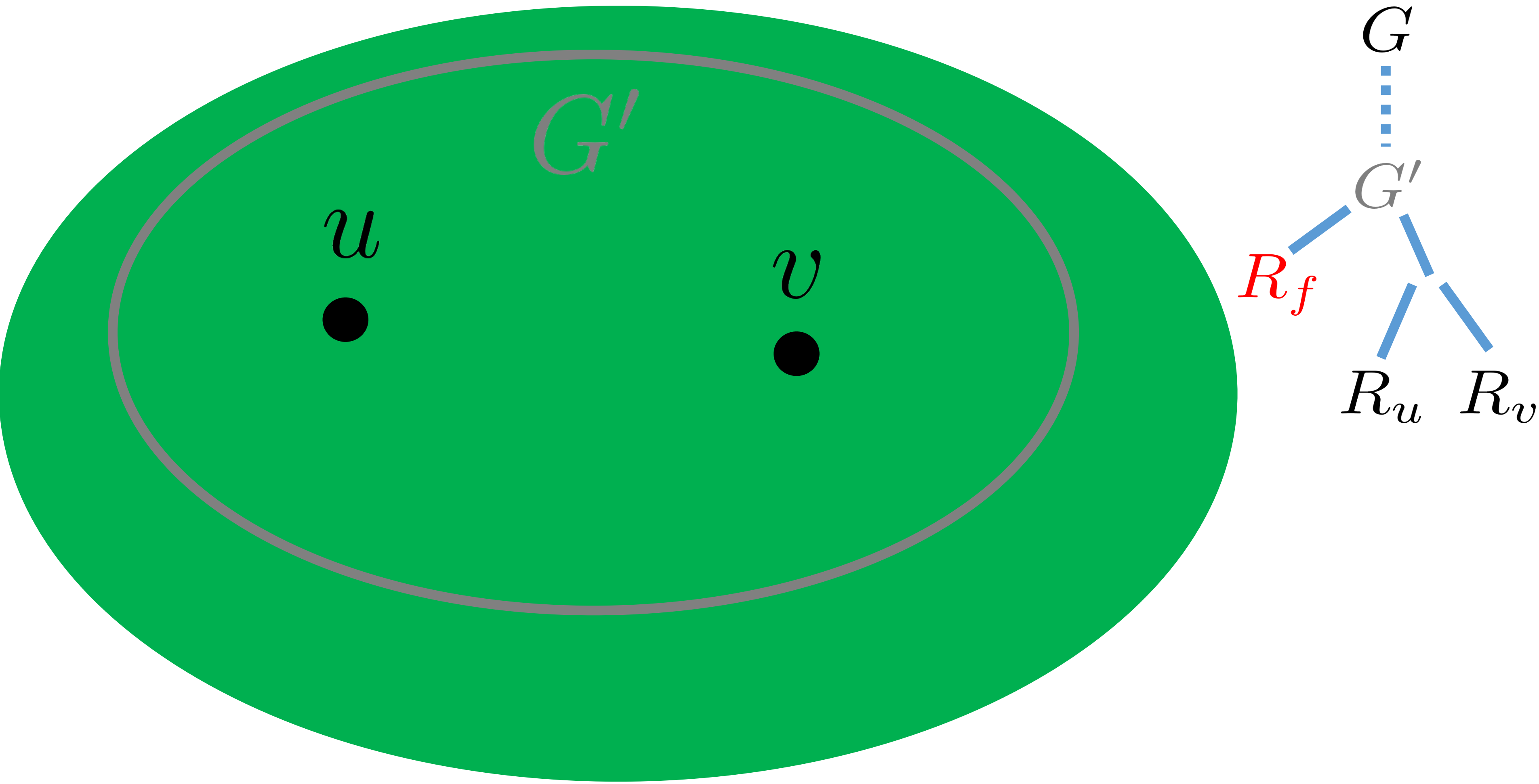
Case 3: Shortest path doesn't touch ∂P for any ancestor piece P of R_f



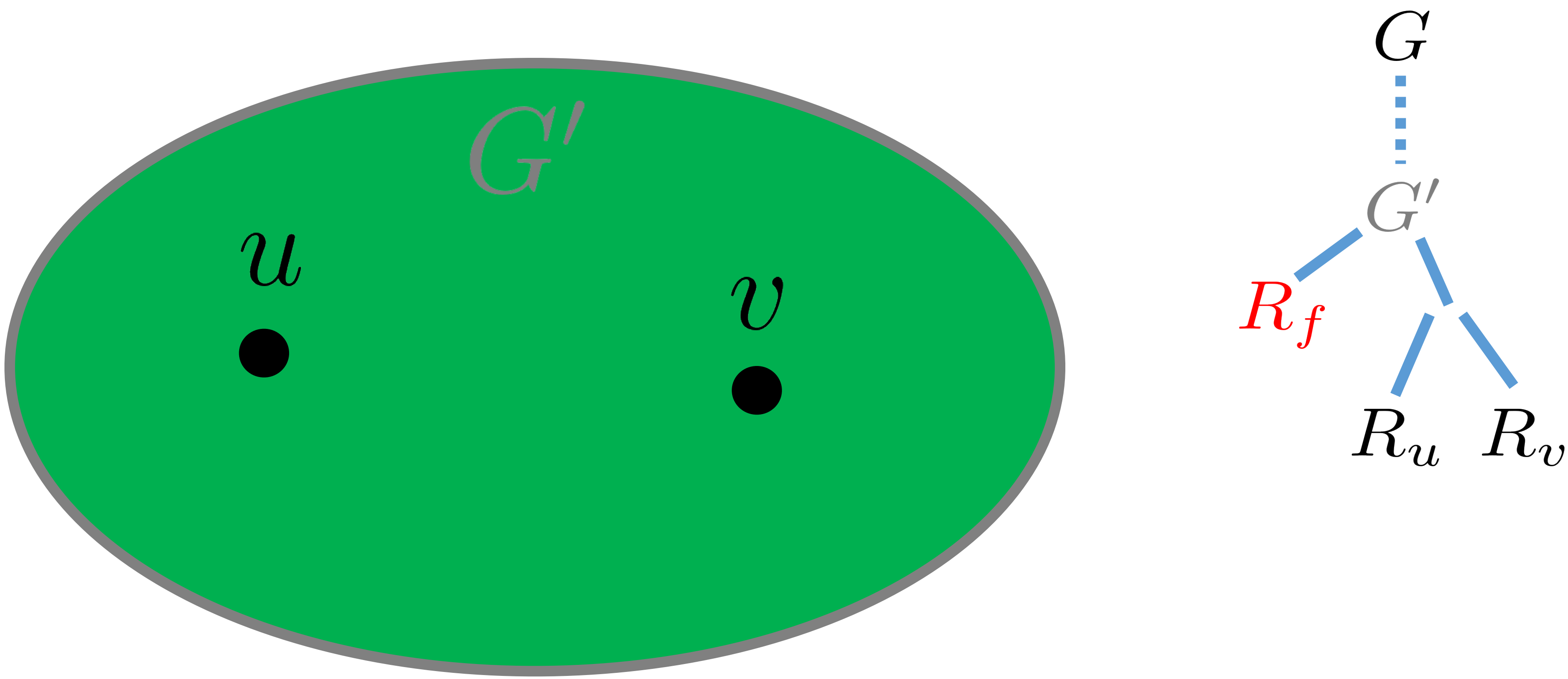
Case 3: Shortest path doesn't touch ∂P for any ancestor piece P of R_f



Case 3: Shortest path doesn't touch ∂P for any ancestor piece P of R_f

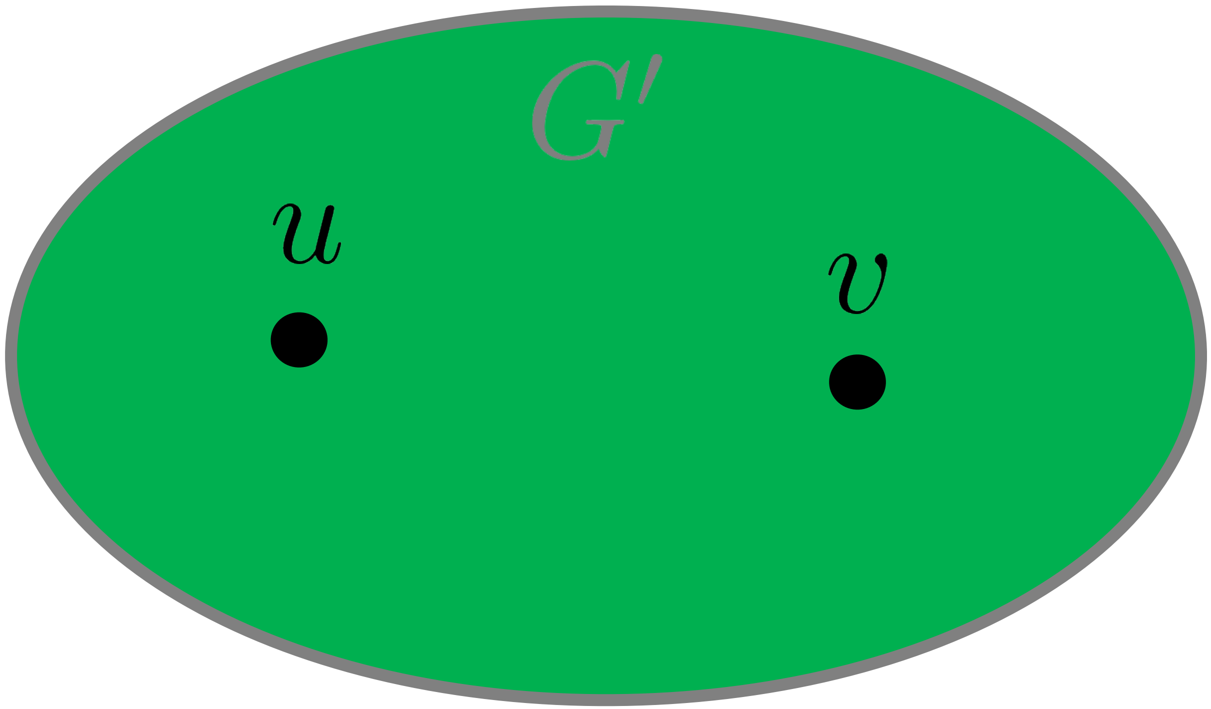


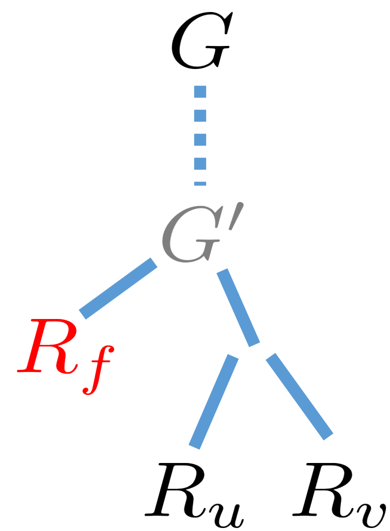
Case 3: Shortest path doesn't touch ∂P for any ancestor piece P of R_f



use labeling without faults in G'

Case 3: Shortest path doesn't touch ∂P for any ancestor piece P of R_f

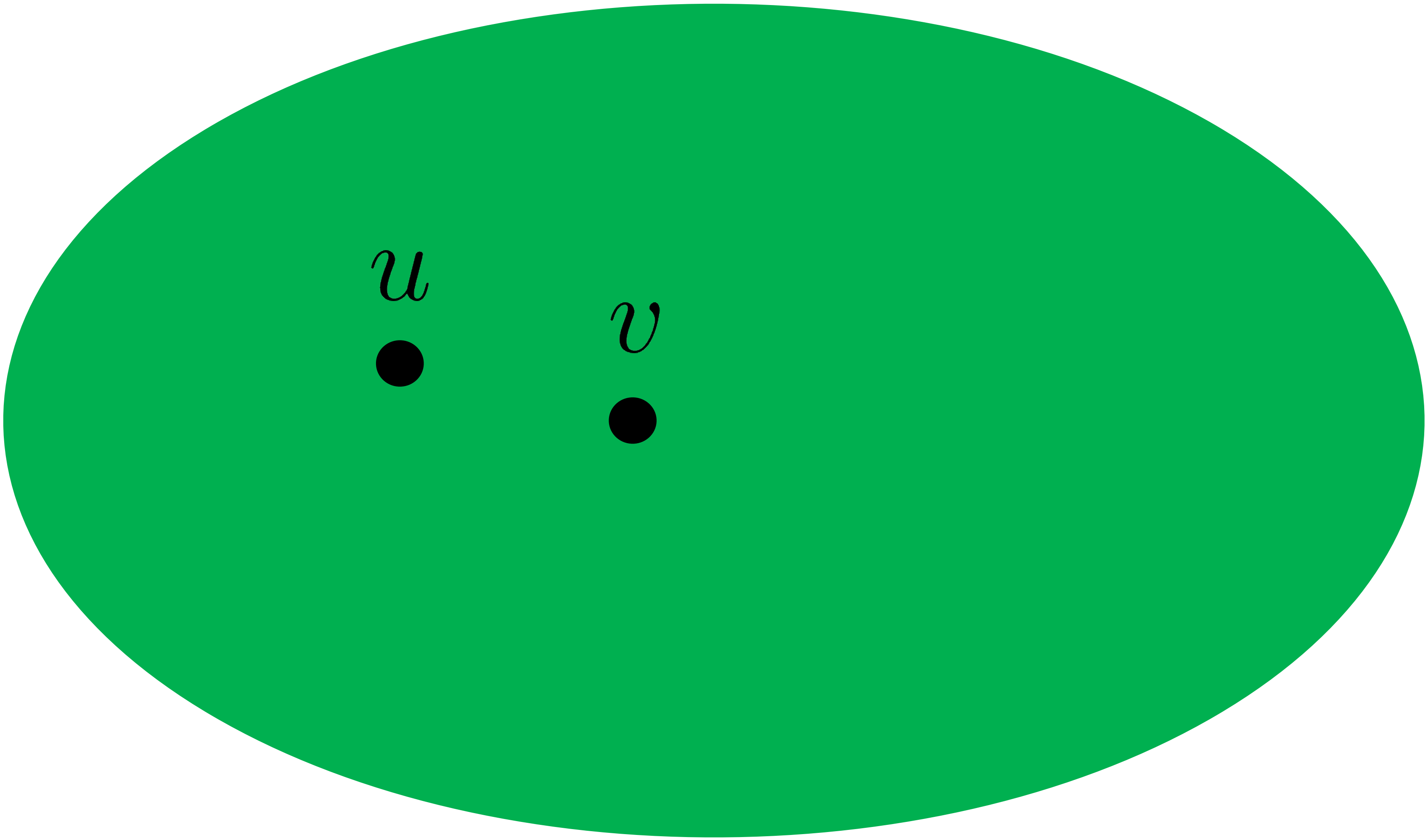




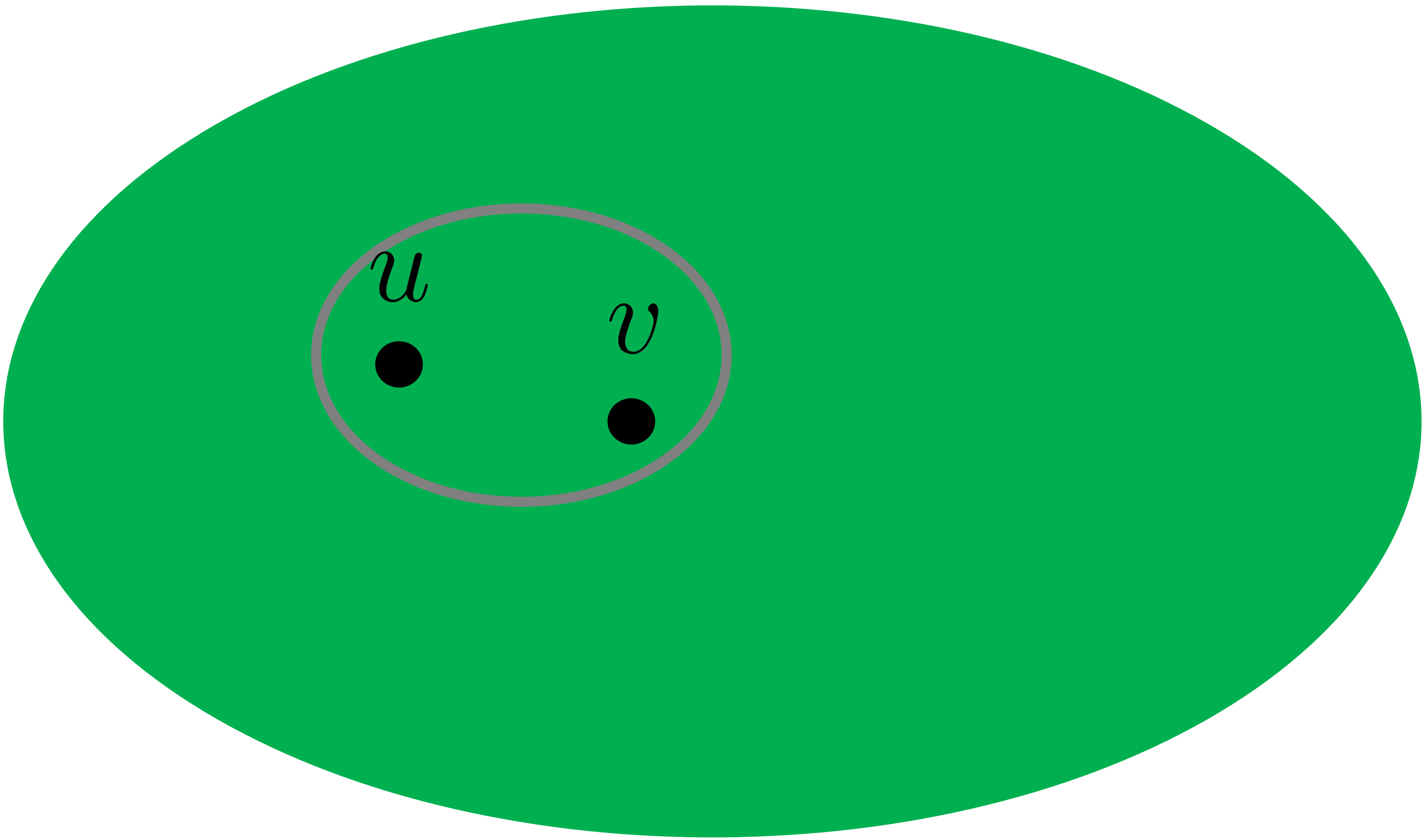
use labeling without faults in G'

$\tilde{O}(\sqrt{n})$ space

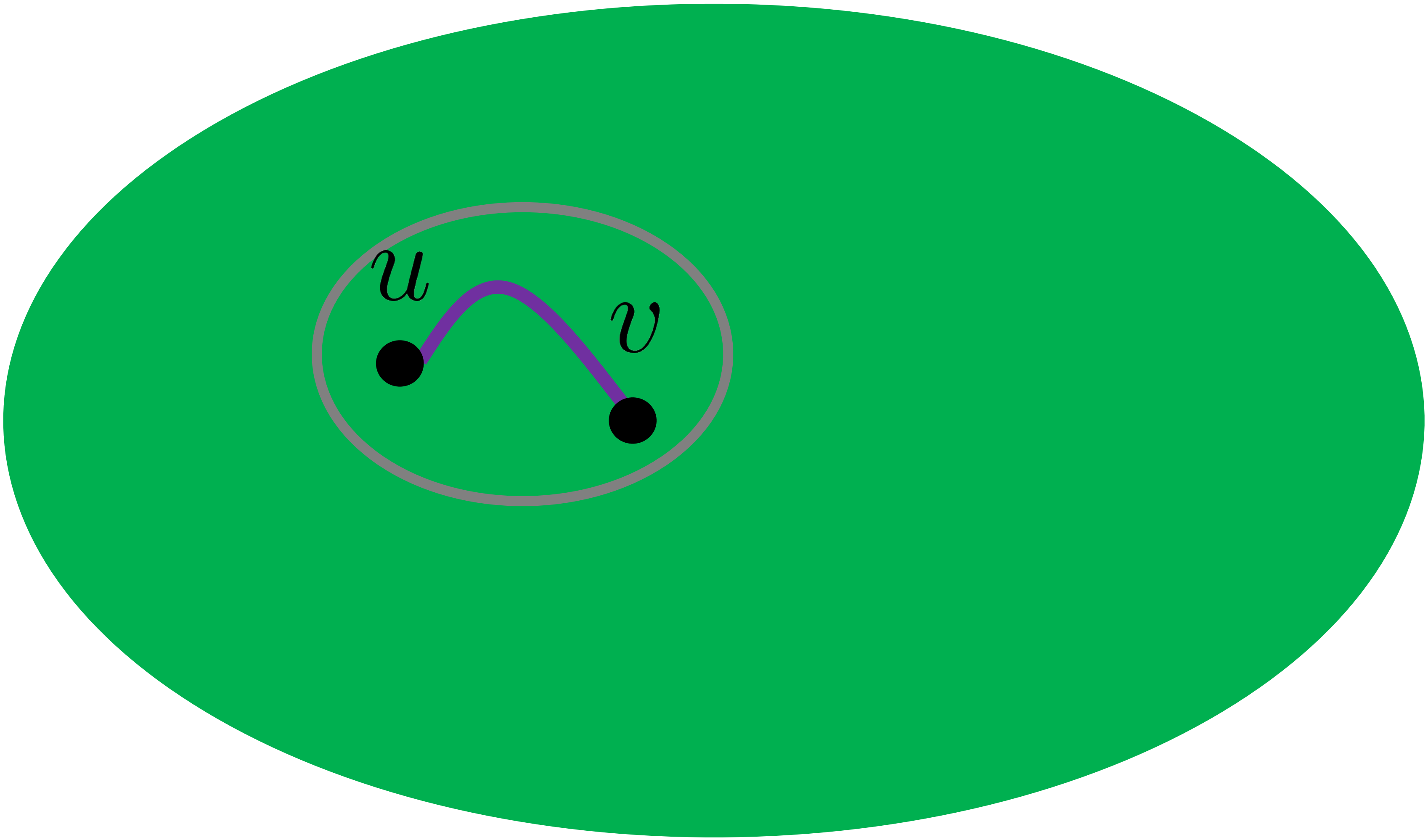
Last case: $R_u = R_v$ and the shortest path doesn't touch ∂R_u



Last case: $R_u = R_v$ and the shortest path doesn't touch ∂R_u

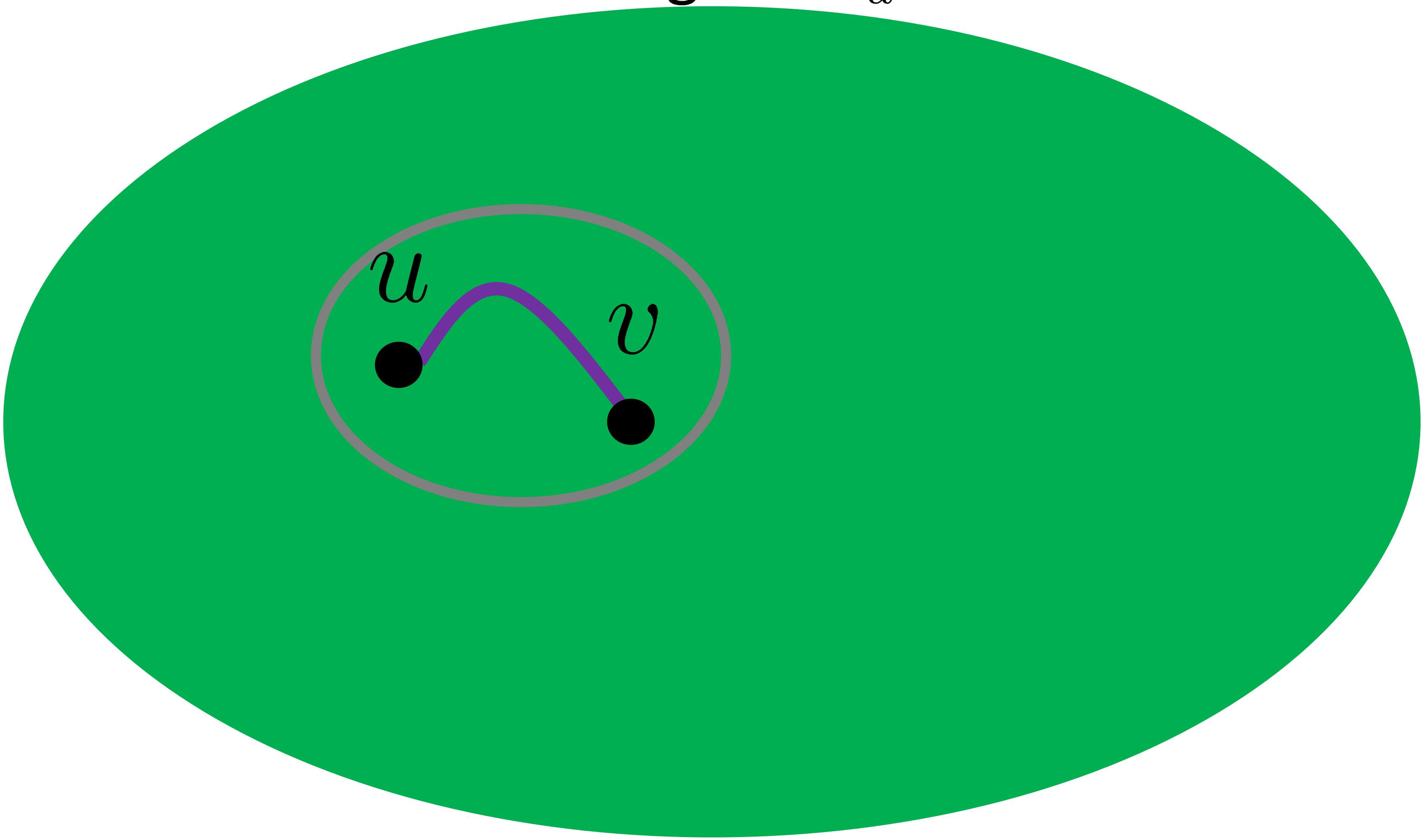


Last case: $R_u = R_v$ and the shortest path doesn't touch ∂R_u



Last case: $R_u = R_v$ and the shortest path doesn't touch ∂R_u

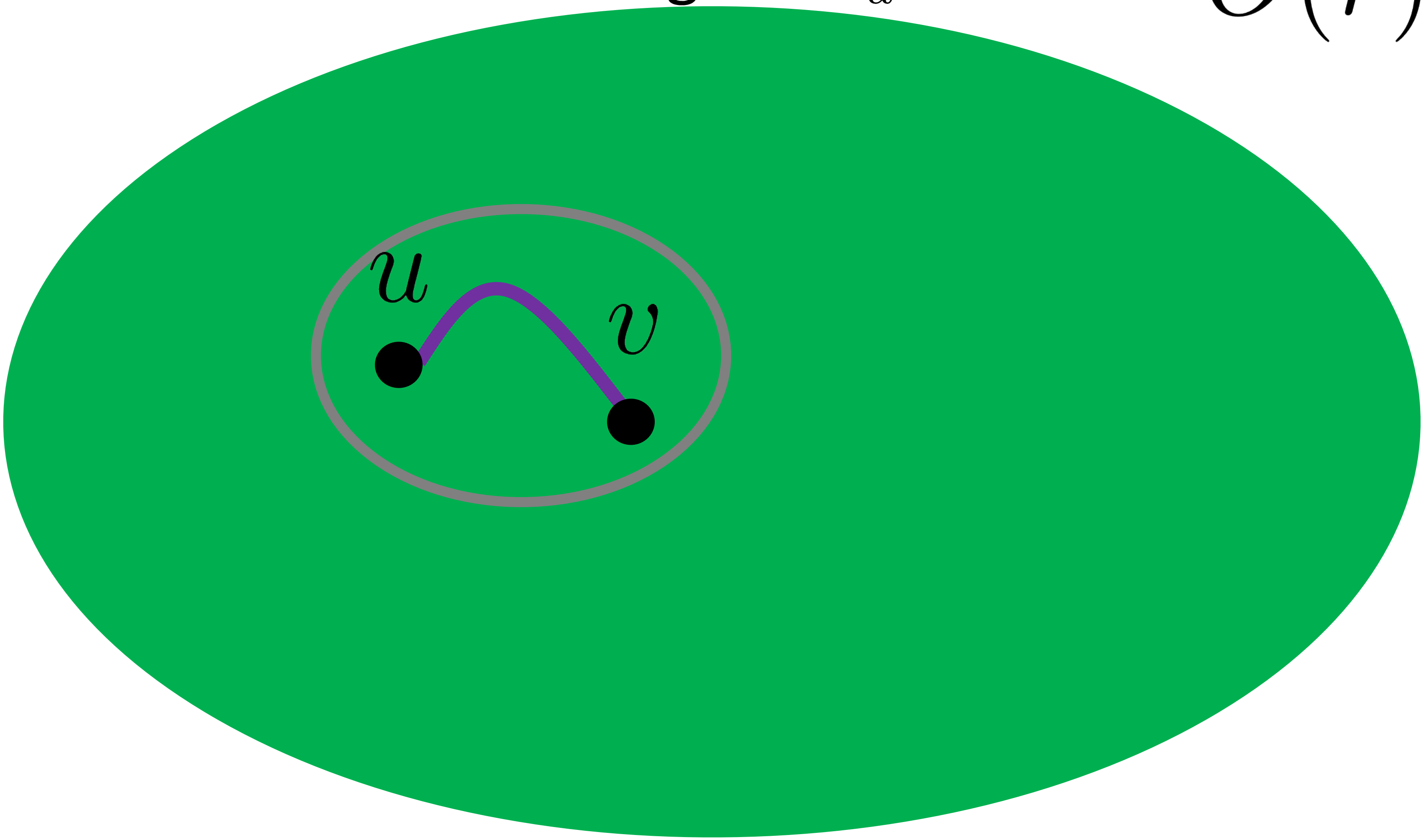
Label stores the entire region R_u



Last case: $R_u = R_v$ and the shortest path doesn't touch ∂R_u

Label stores the entire region R_u

$\tilde{O}(r)$



Choose r:

Choose r:

$$\tilde{O}(n/\sqrt{r} + r + \sqrt{n})$$

Choose r:

$$\tilde{O}(n/\sqrt{r} + r + \sqrt{n})$$

$$r = n^{2/3}$$

Choose r:

$$\tilde{O}(n/\sqrt{r} + r + \sqrt{n})$$

$$r = n^{2/3}$$

$$\tilde{O}(n^{2/3})$$

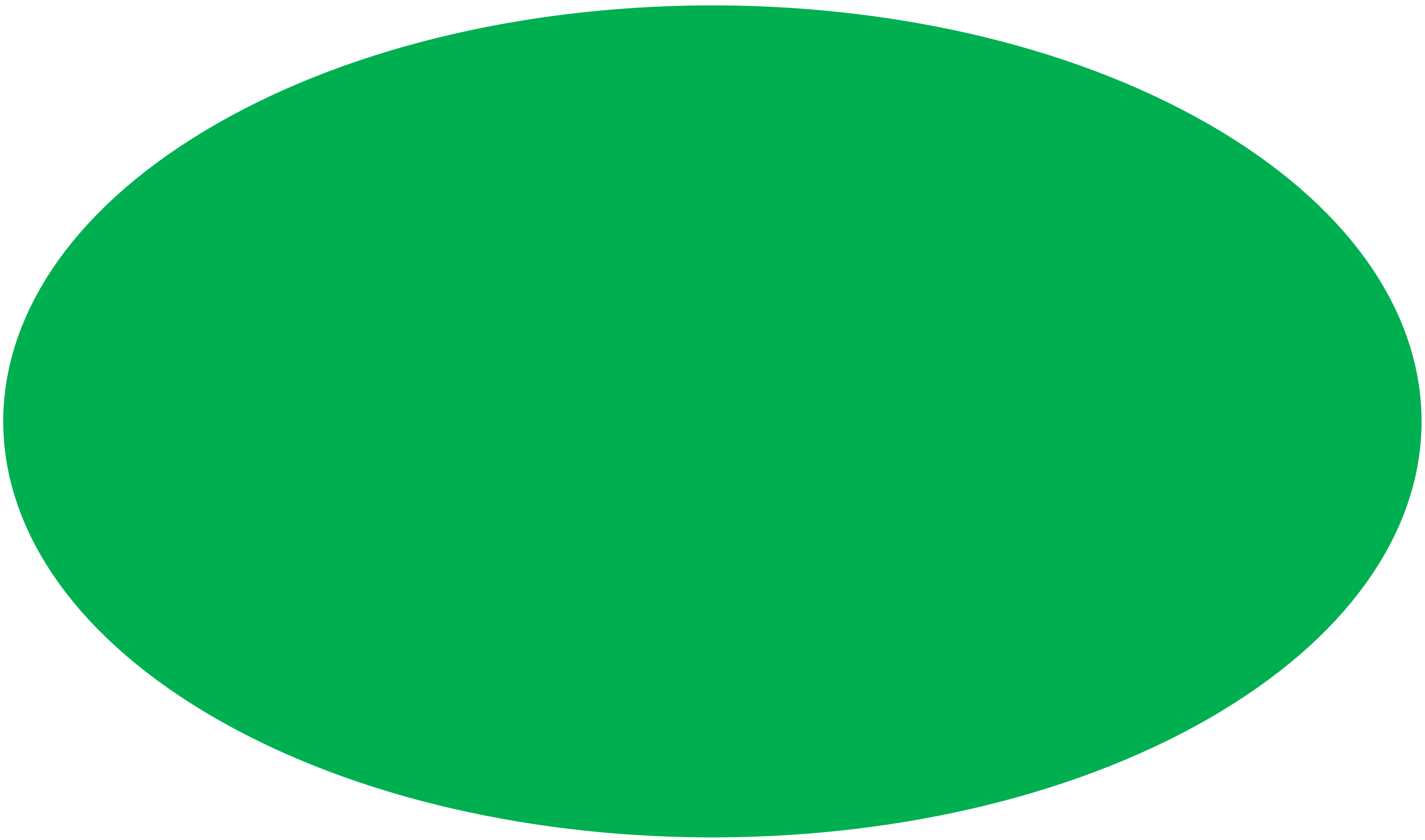
Choose r:

$$\tilde{O}(n/\sqrt{r} + r + \sqrt{n})$$

$$r = n^{2/3}$$

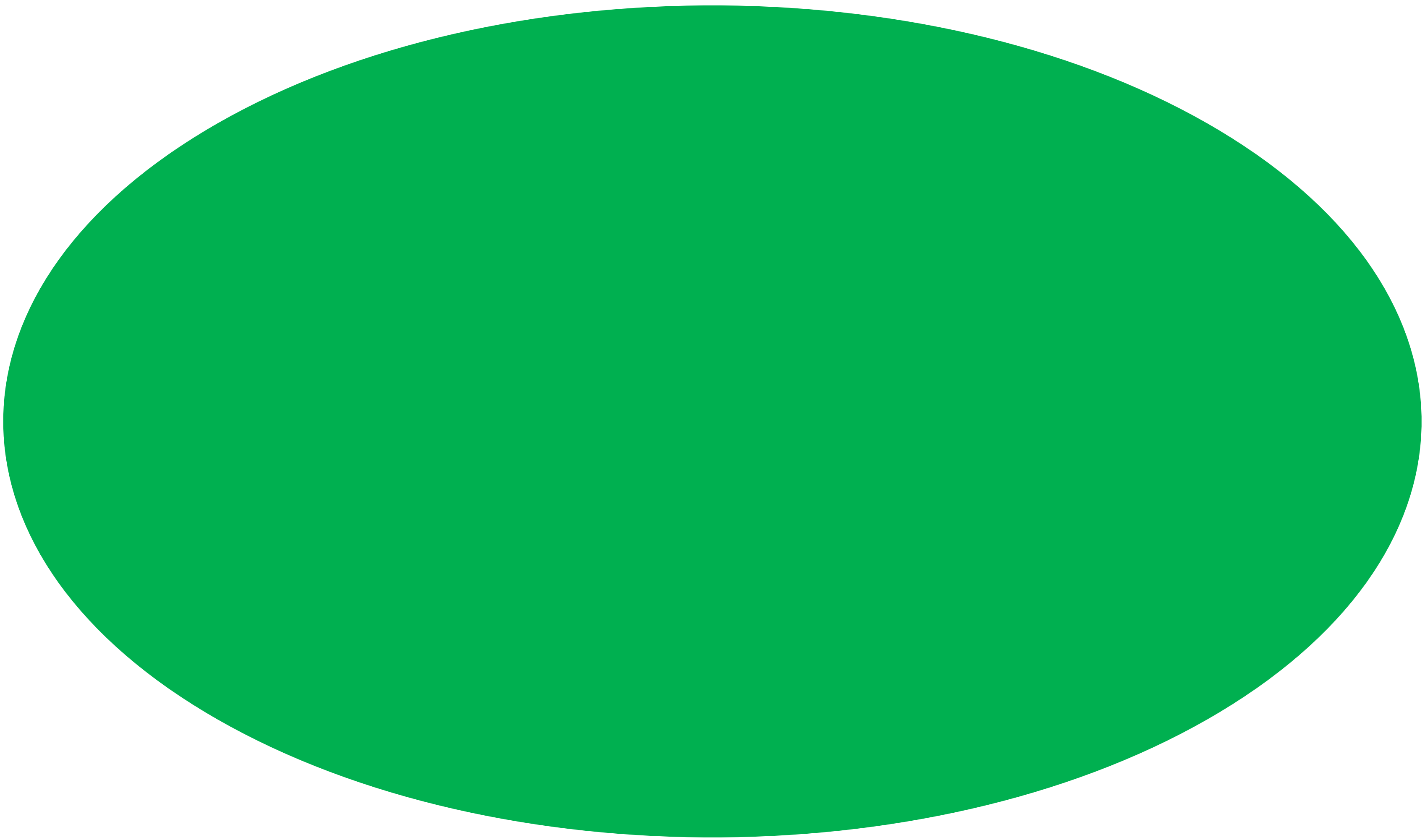
$$\tilde{O}(n^{2/3})$$

More carefully…



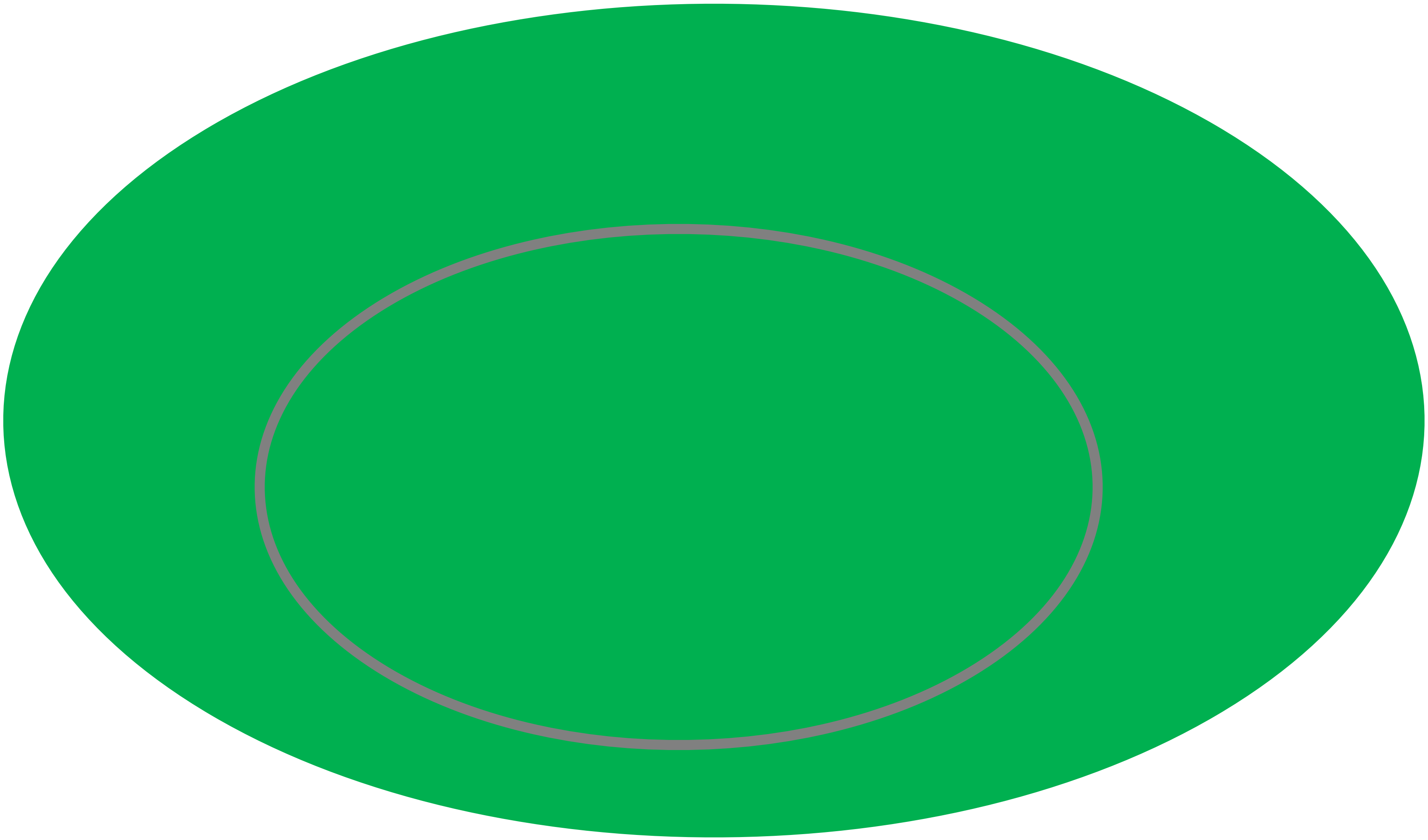
More carefully…

a vertex can belong to many seperators



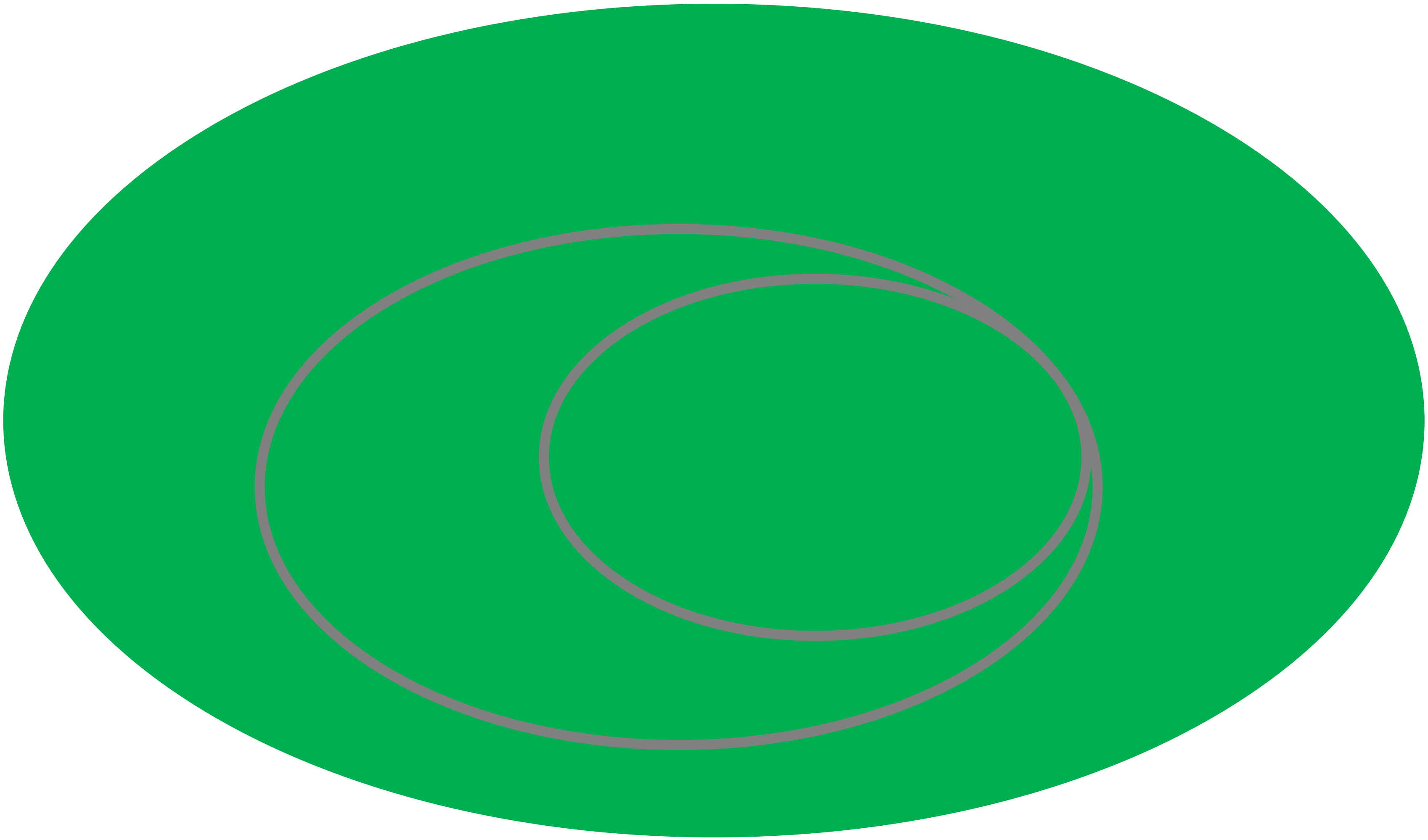
More carefully…

a vertex can belong to many seperators



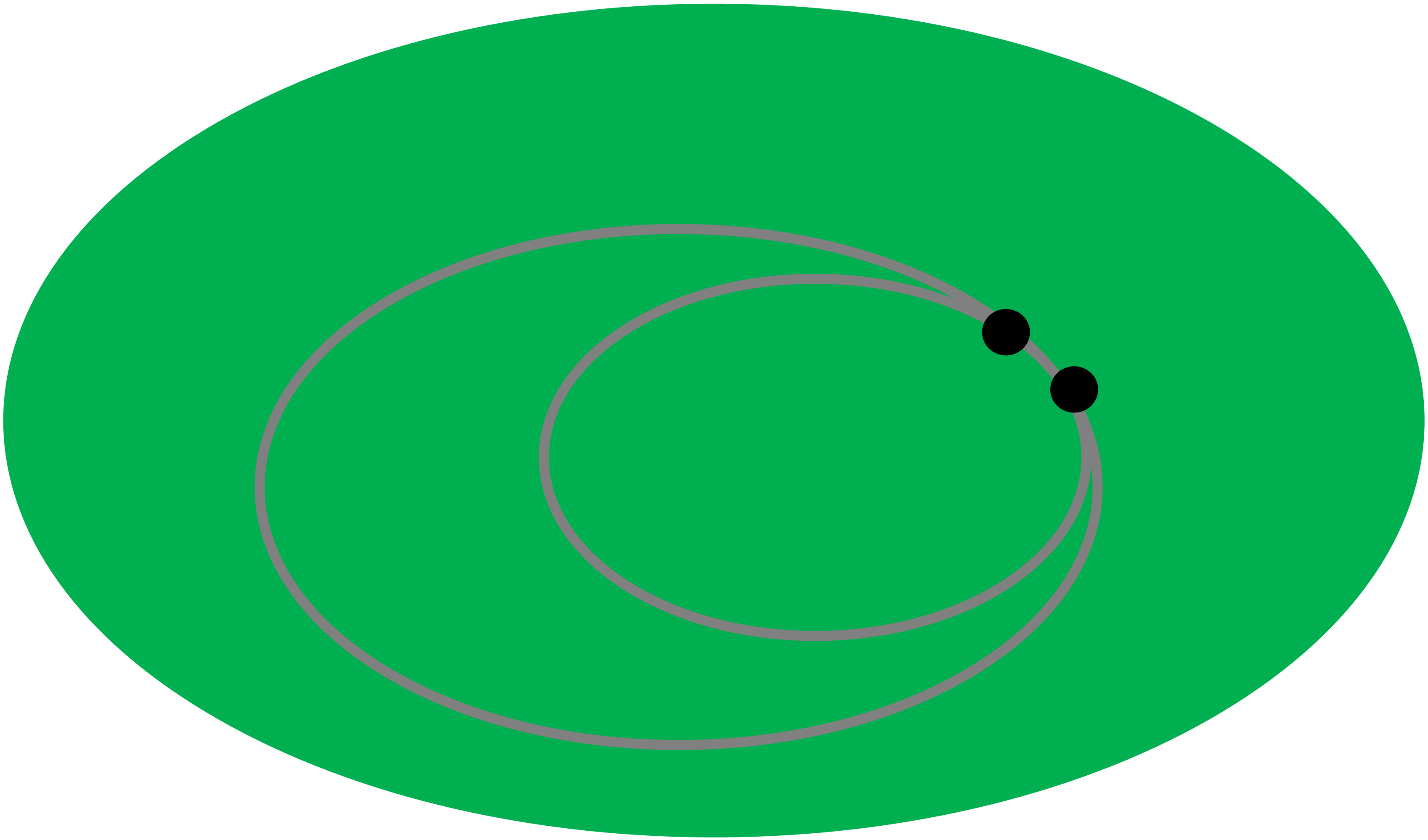
More carefully…

a vertex can belong to many seperators



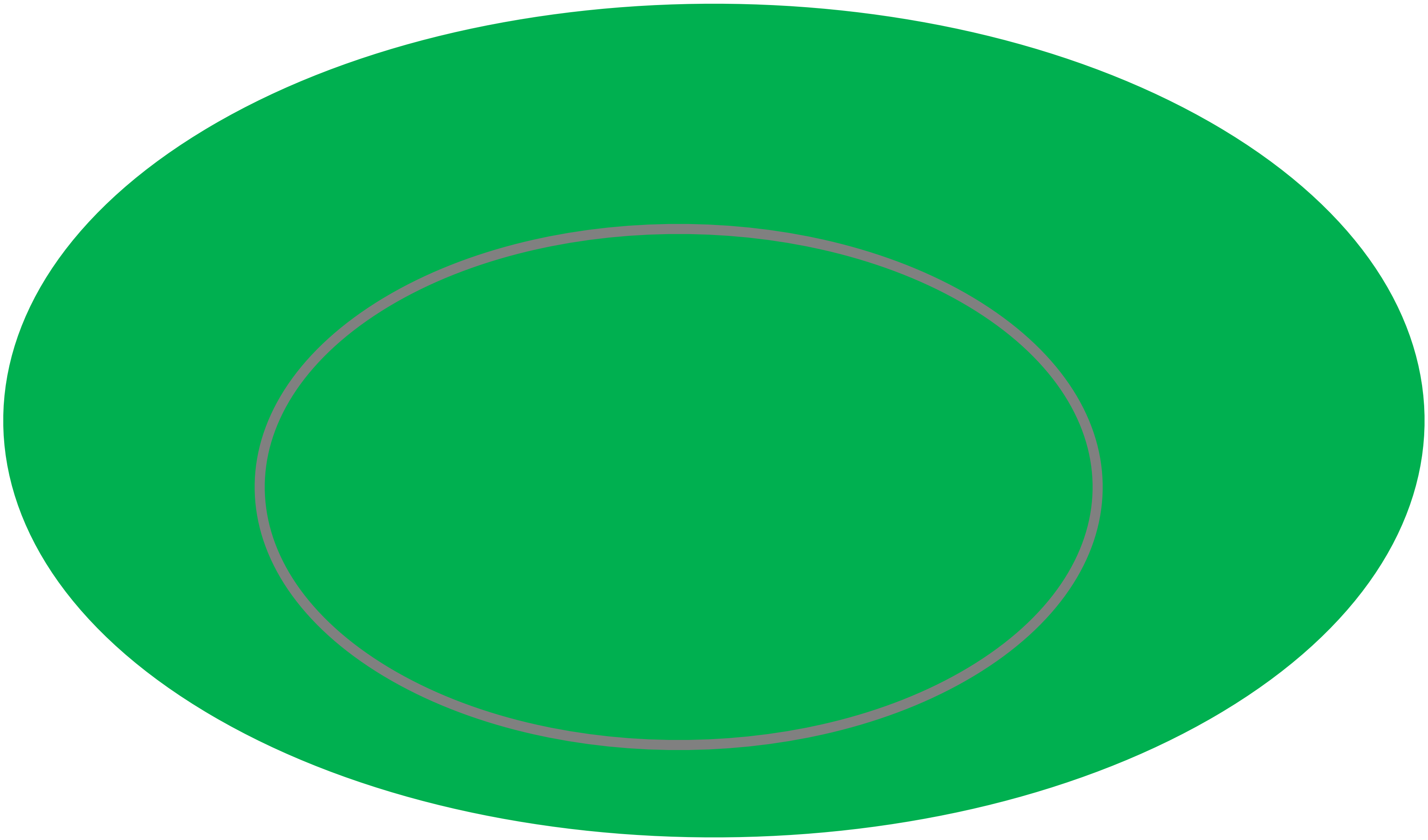
More carefully...

a vertex can belong to many seperators



More carefully…

a vertex can belong to many seperators



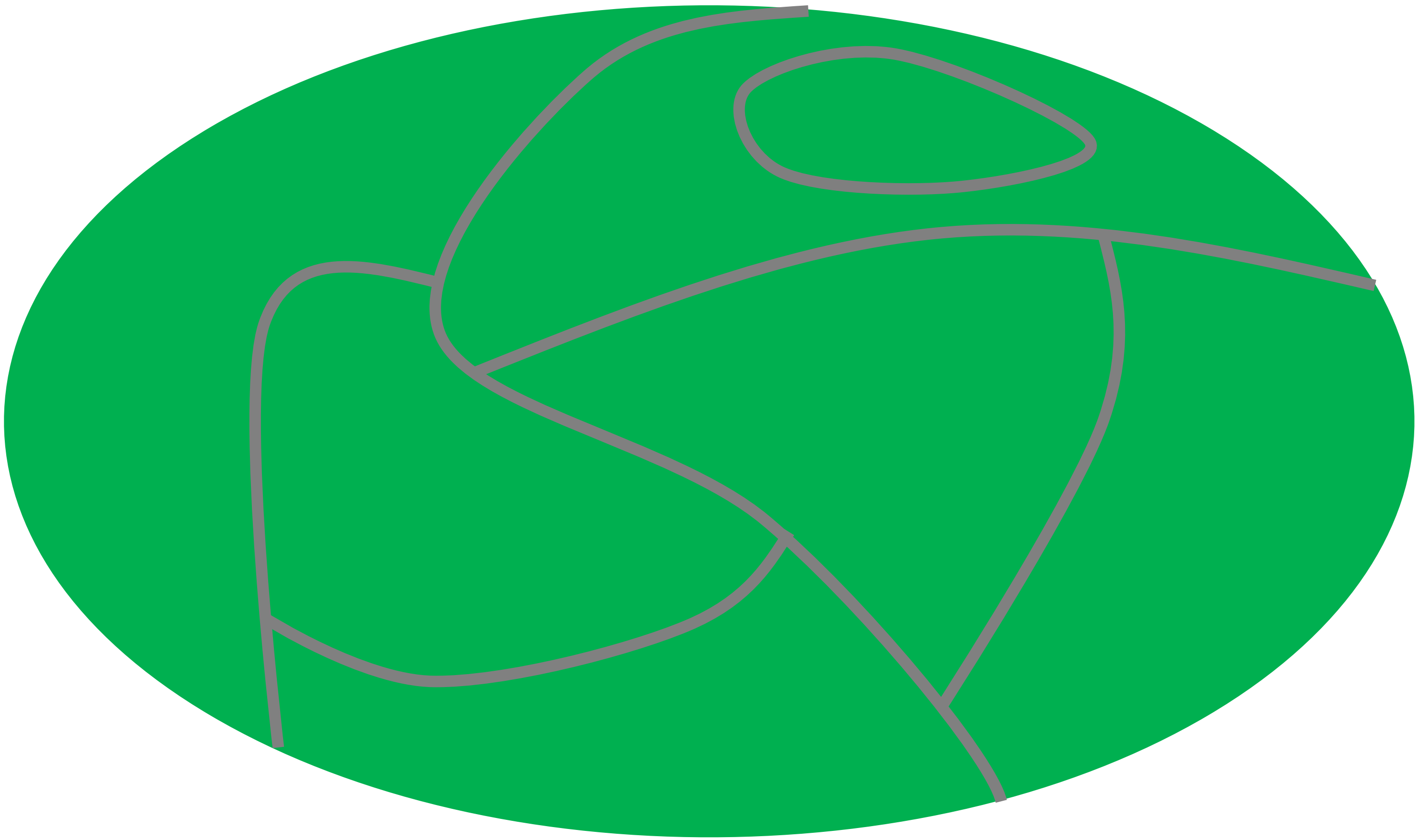
More carefully…

More carefully…

There might be holes inside pieces

More carefully…

There might be holes inside pieces



More carefully…

More carefully…

Query time is only $\tilde{O}(\sqrt{n})$

More carefully…

Query time is only $\tilde{O}(\sqrt{n})$

Open problems

Open problems

- Non-trivial (sublinear) labeling in the case of 2 faults?

Open problems

- Non-trivial (sublinear) labeling in the case of 2 faults?

  Current label stores the entire graph!

Open problems

- Non-trivial (sublinear) labeling in the case of 2 faults?

  Current label stores the entire graph!

- Lower bounds for fault tolerant setting?

A note about lower bounds

[Gavoille et al. SODA '01]

general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

A note about lower bounds

[Gavoille et al. SODA '01]

general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices

A note about lower bounds

[Gavoille et al. SODA '01]

general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices

A note about lower bounds

[Gavoille et al. SODA '01]

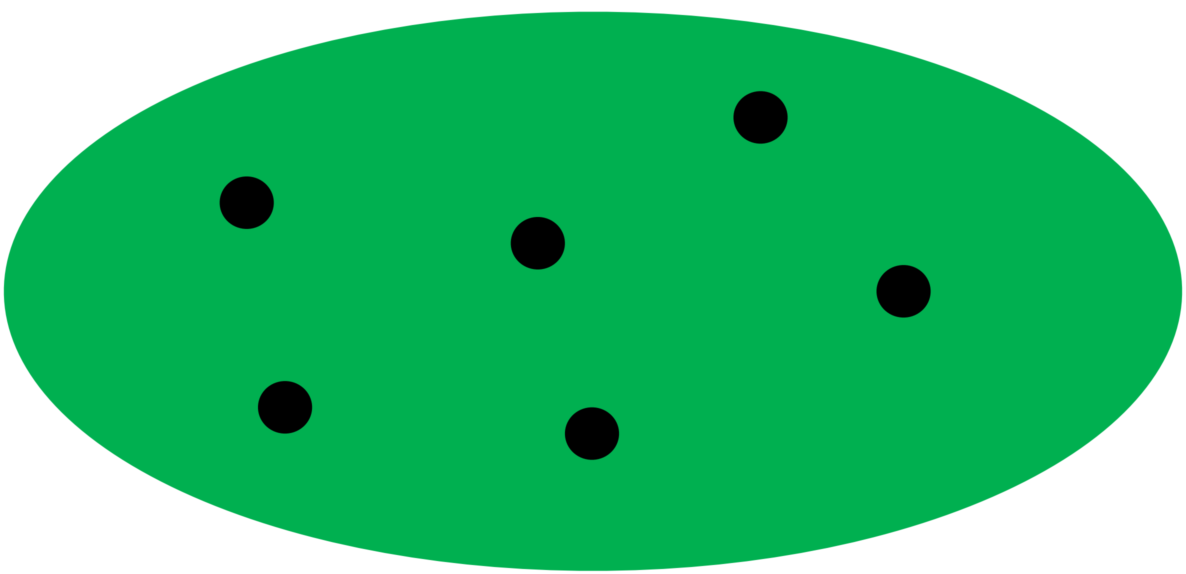
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



A note about lower bounds

[Gavoille et al. SODA '01]

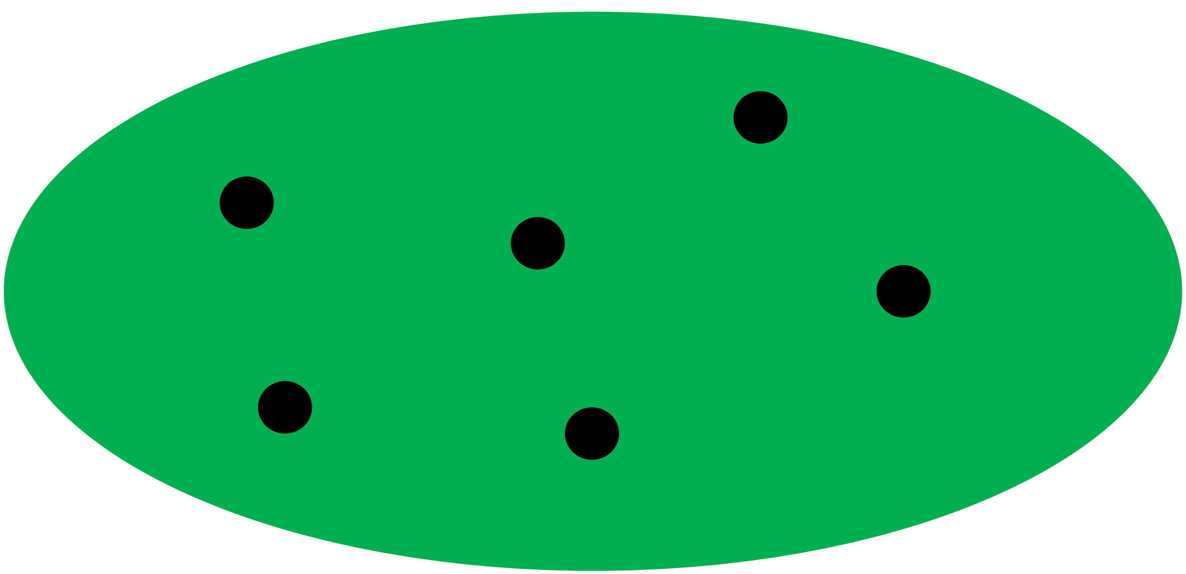
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

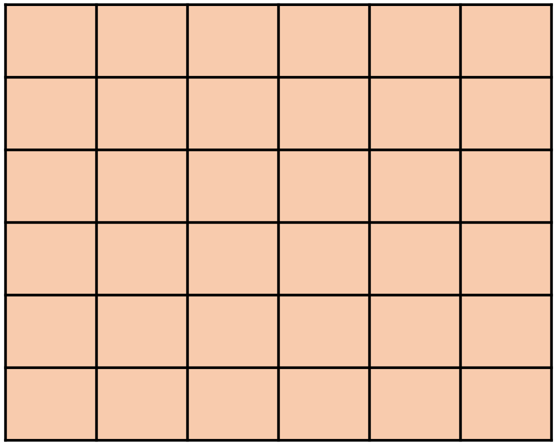
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



k by k distance matrix



A note about lower bounds

[Gavoille et al. SODA '01]

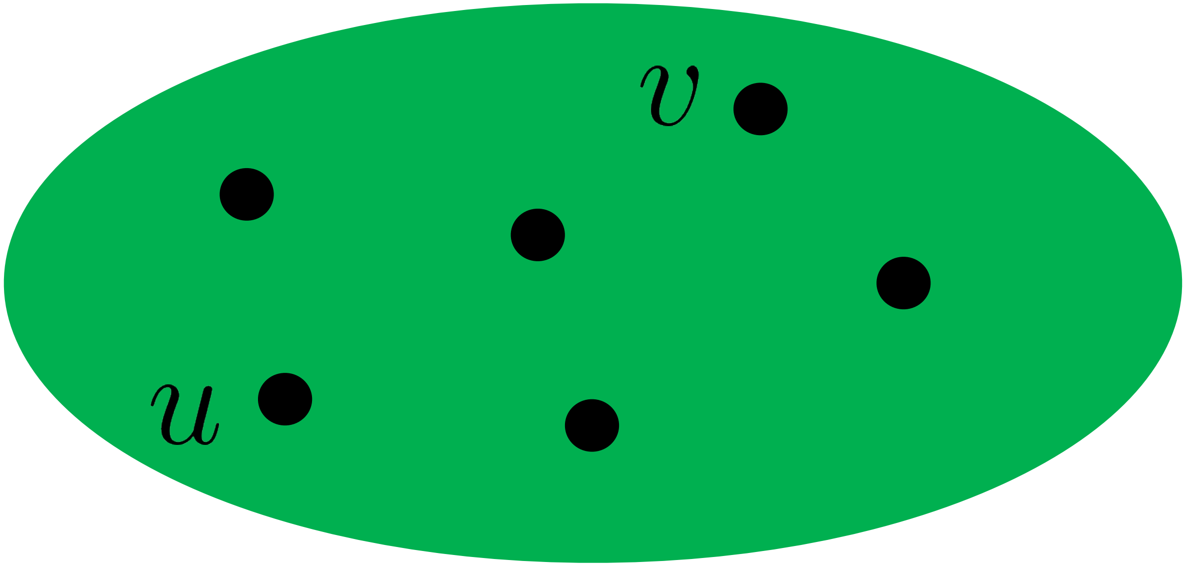
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

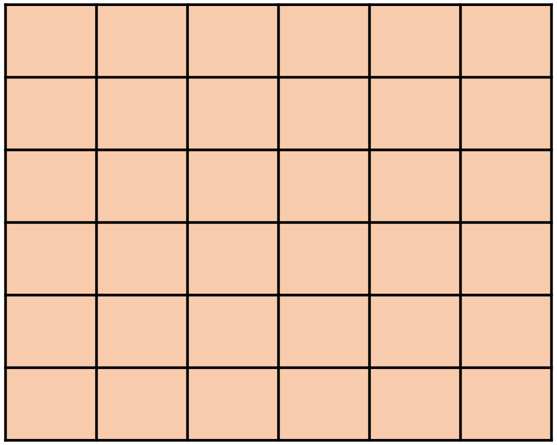
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



k by k distance matrix



A note about lower bounds

[Gavoille et al. SODA '01]

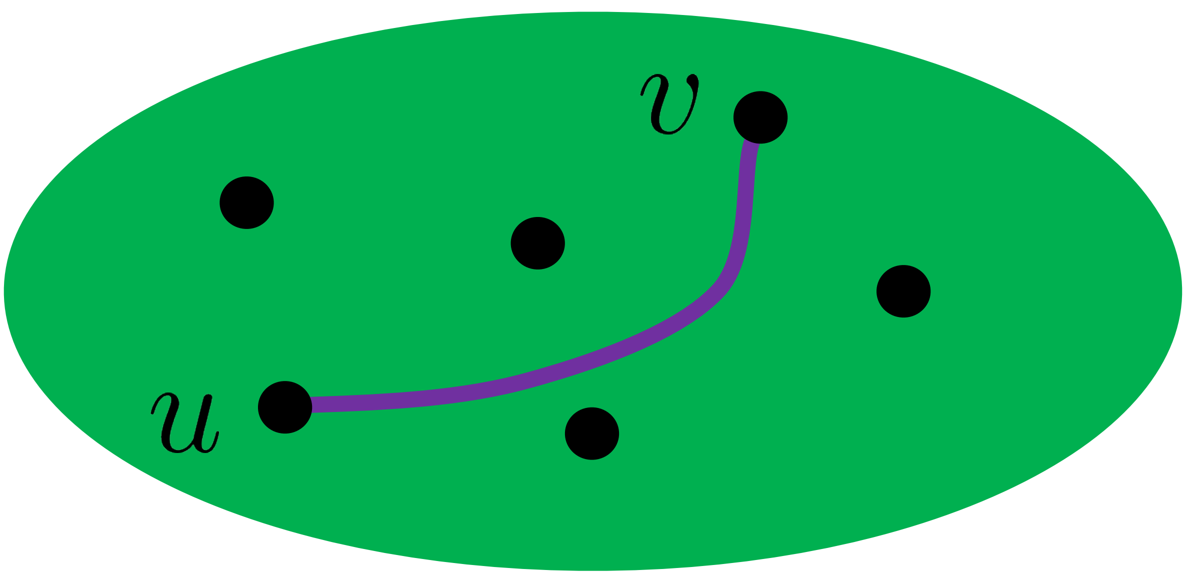
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

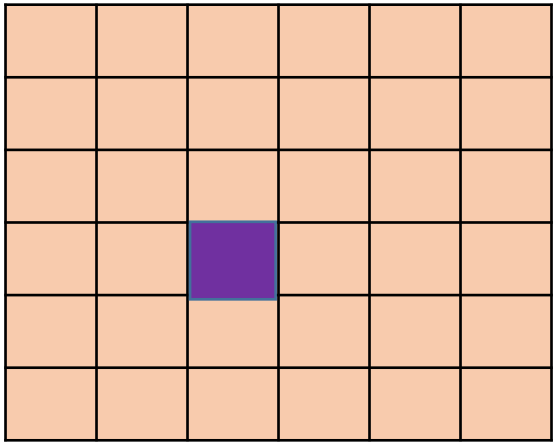
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



k by k distance matrix



A note about lower bounds

[Gavoille et al. SODA '01]

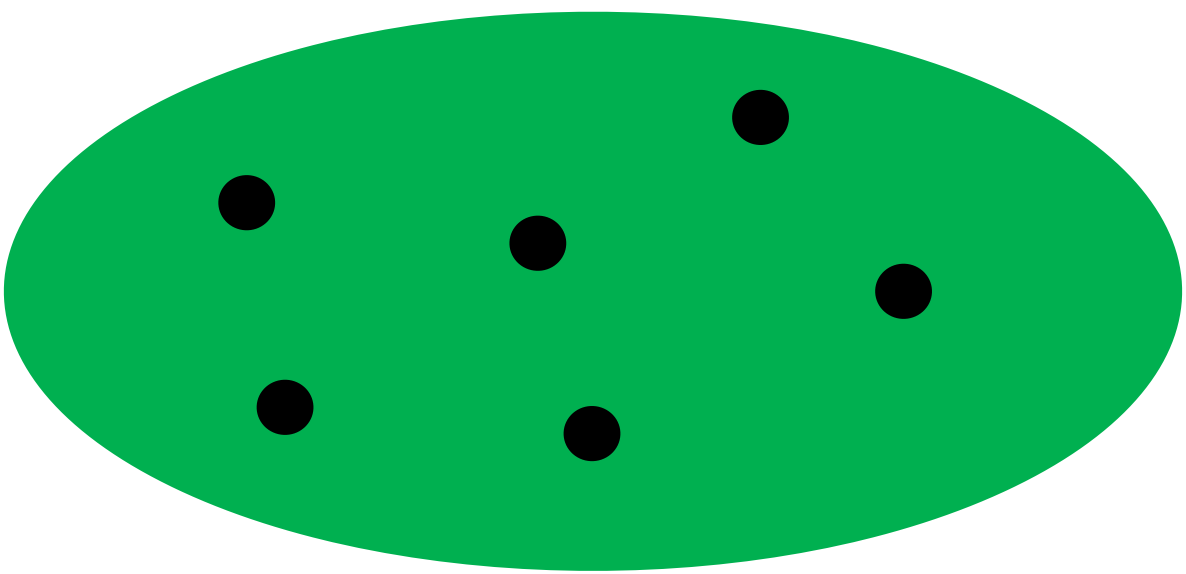
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

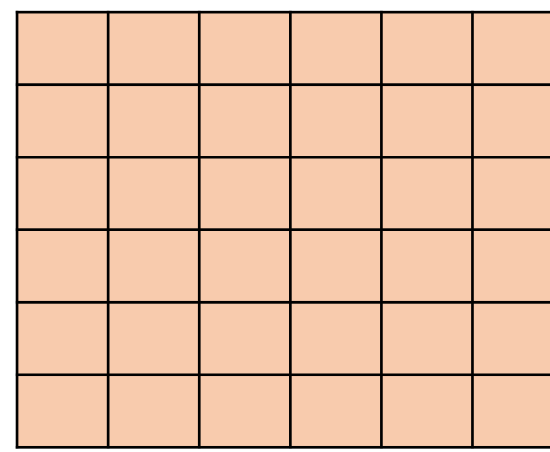
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



k by k distance matrix



A note about lower bounds

[Gavoille et al. SODA '01]

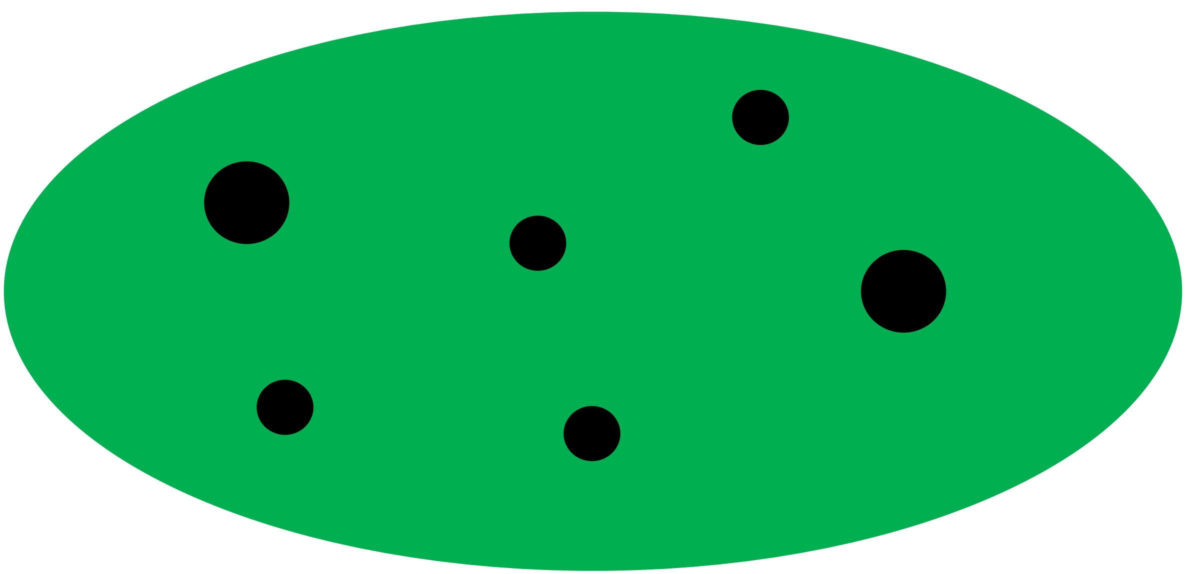
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

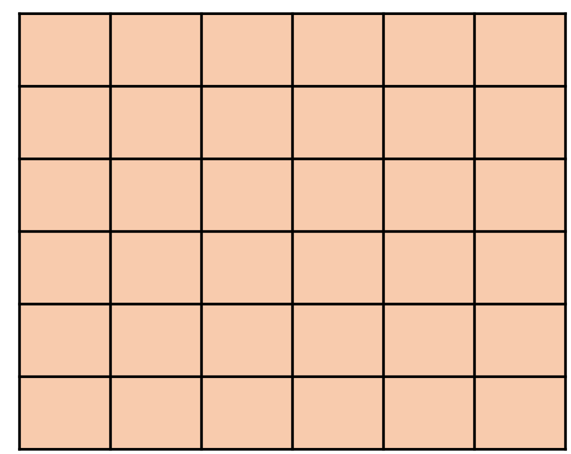
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



k by k distance matrix



A note about lower bounds

[Gavoille et al. SODA '01]

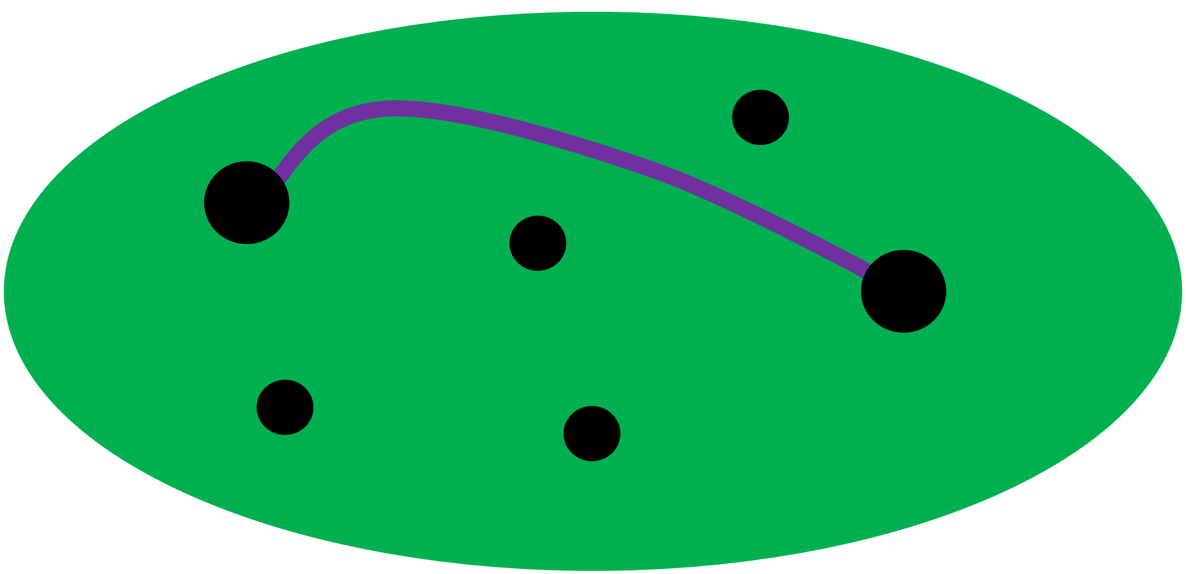
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

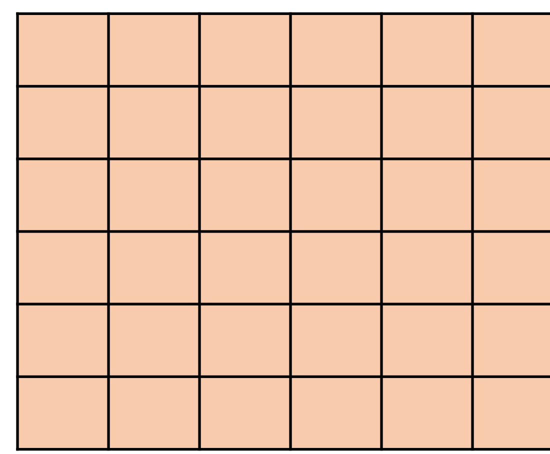
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



k by k distance matrix



A note about lower bounds

[Gavoille et al. SODA '01]

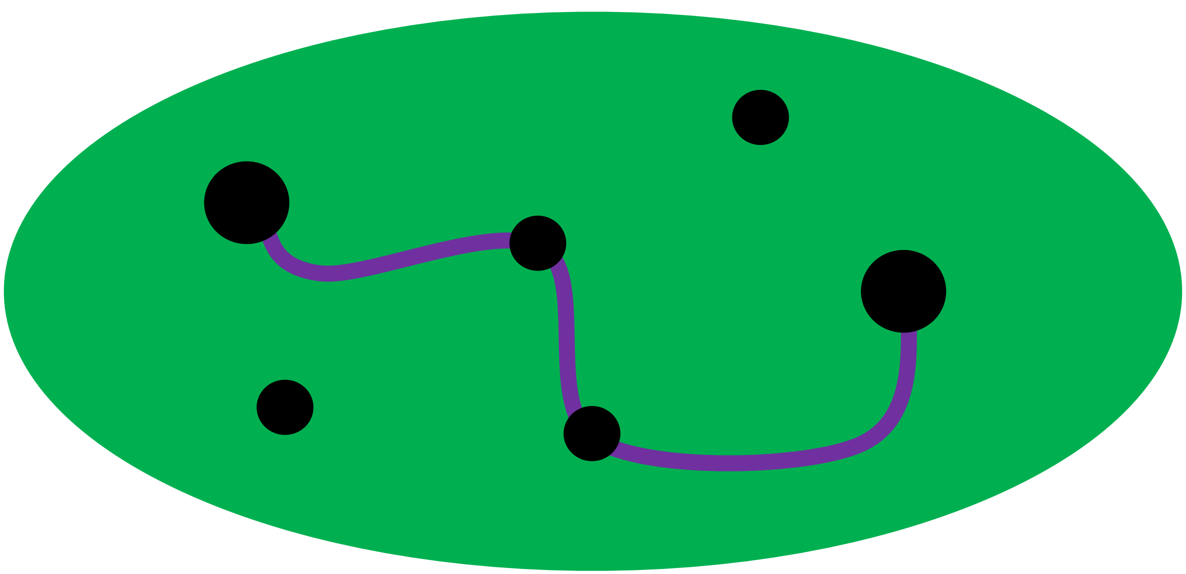
general argument:

k labels encode d data bits $\rightarrow \Omega(\frac{d}{k})$ label size

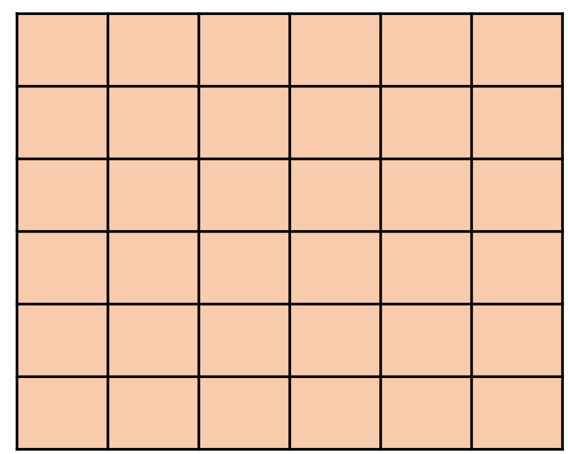
How about $\omega(\sqrt{n})$ lower bound for fault tolerant?

Argument doesn't work!

$\tilde{O}(k^2)$ information bits can solve fault tolerant for k vertices



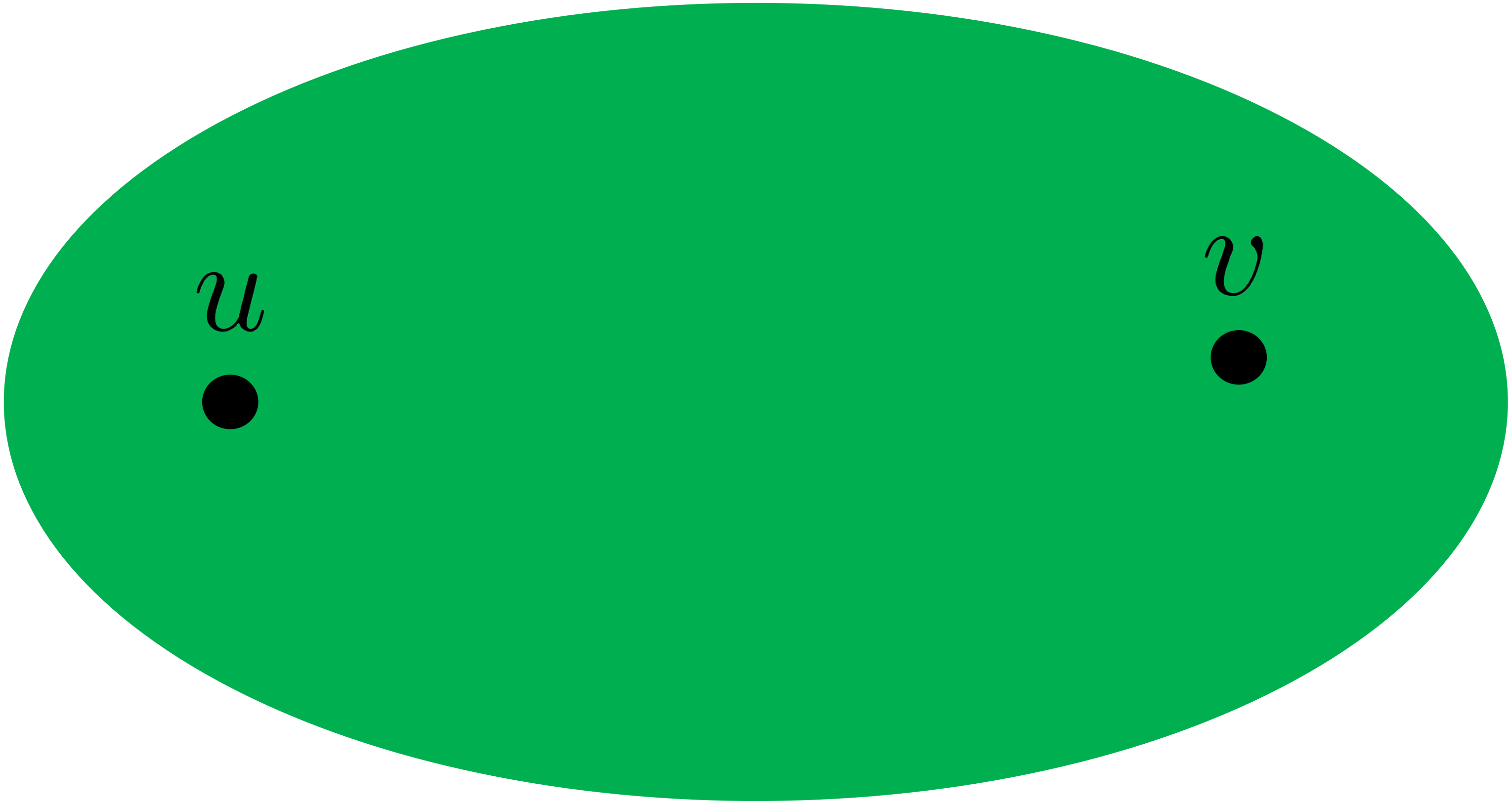
k by k distance matrix



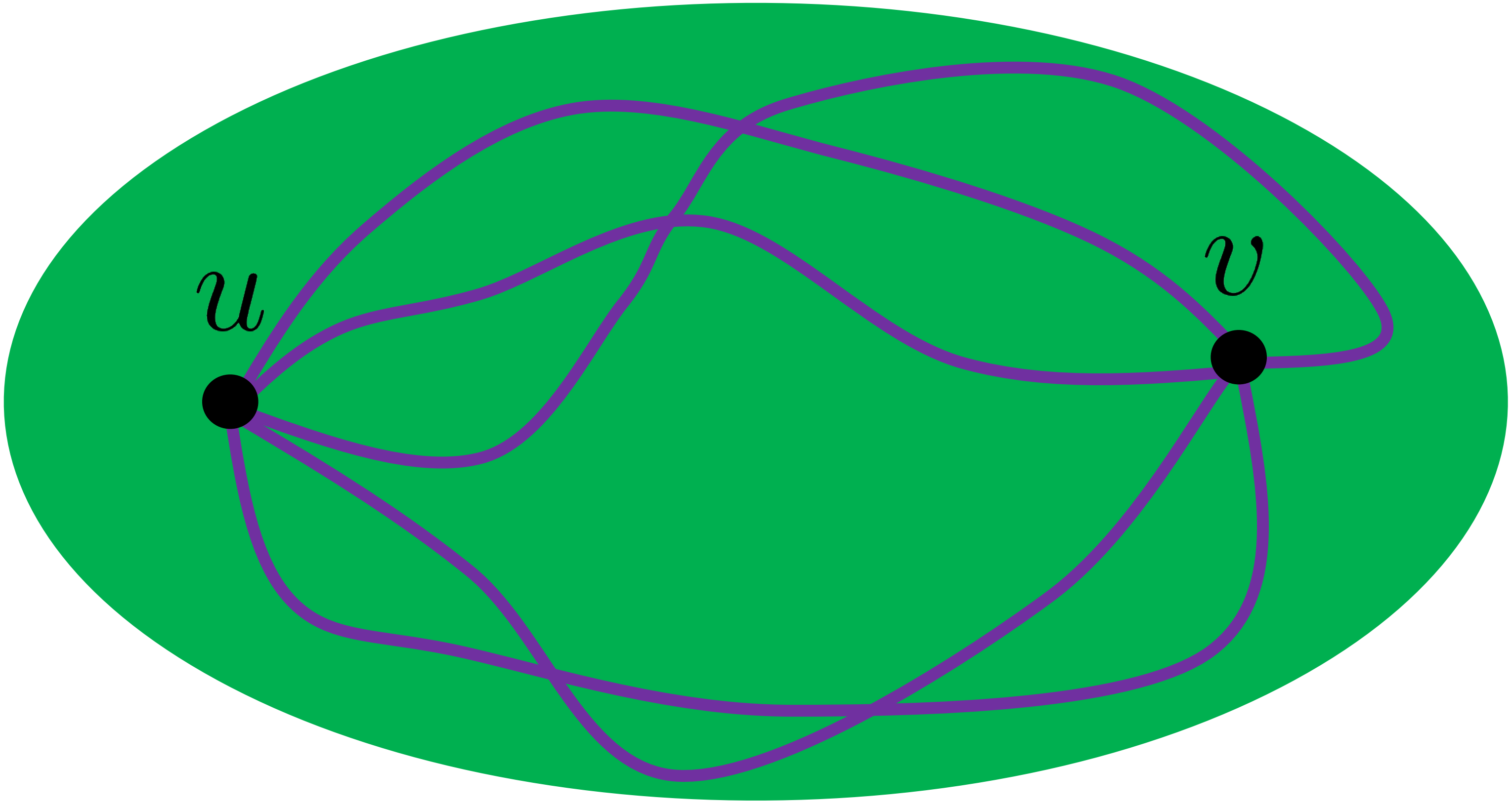
Counting shortest paths

Labels can be extended for counting the number of shortest paths with the same label size up to polylog factors

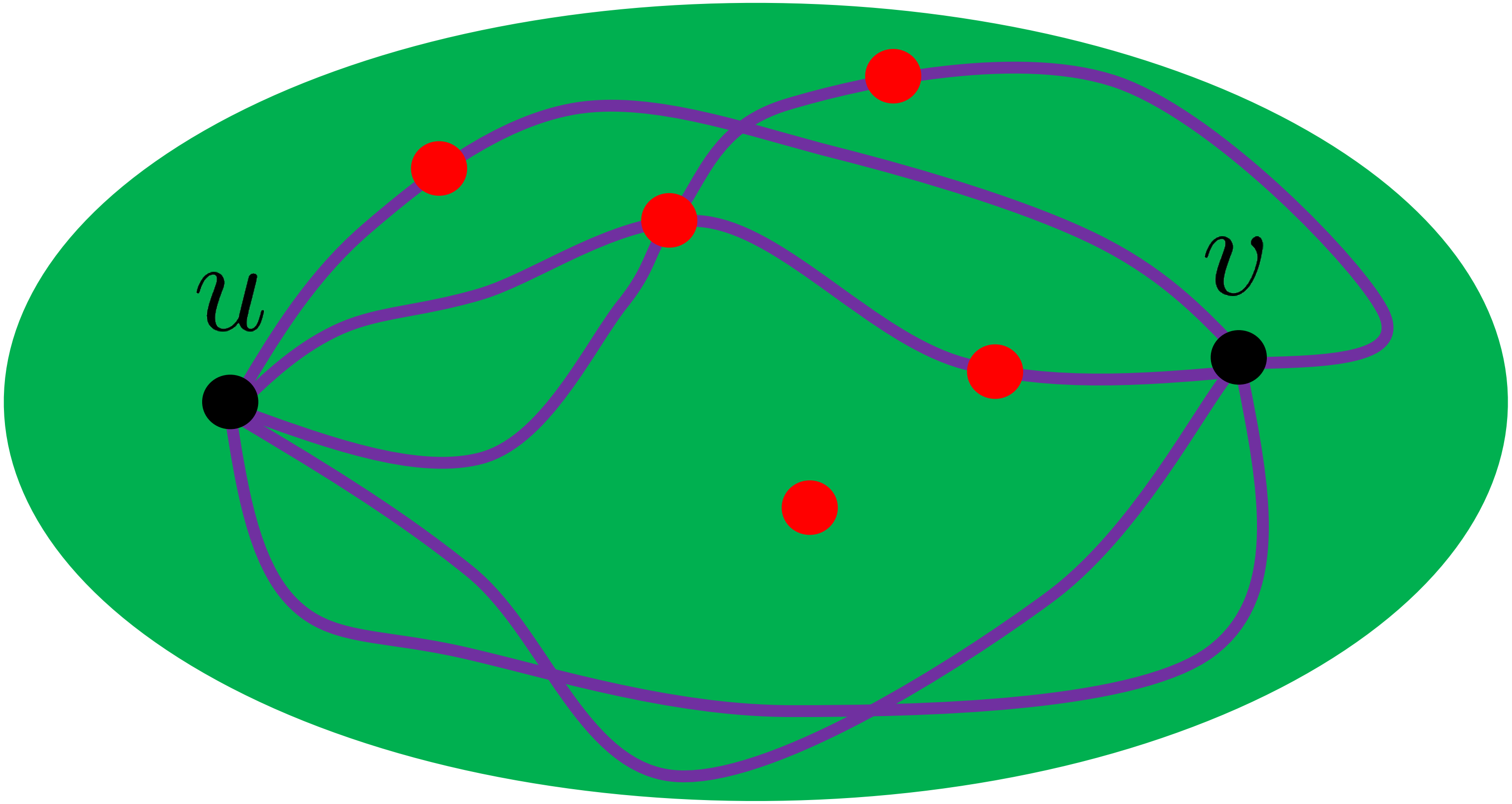
Counting shortest paths



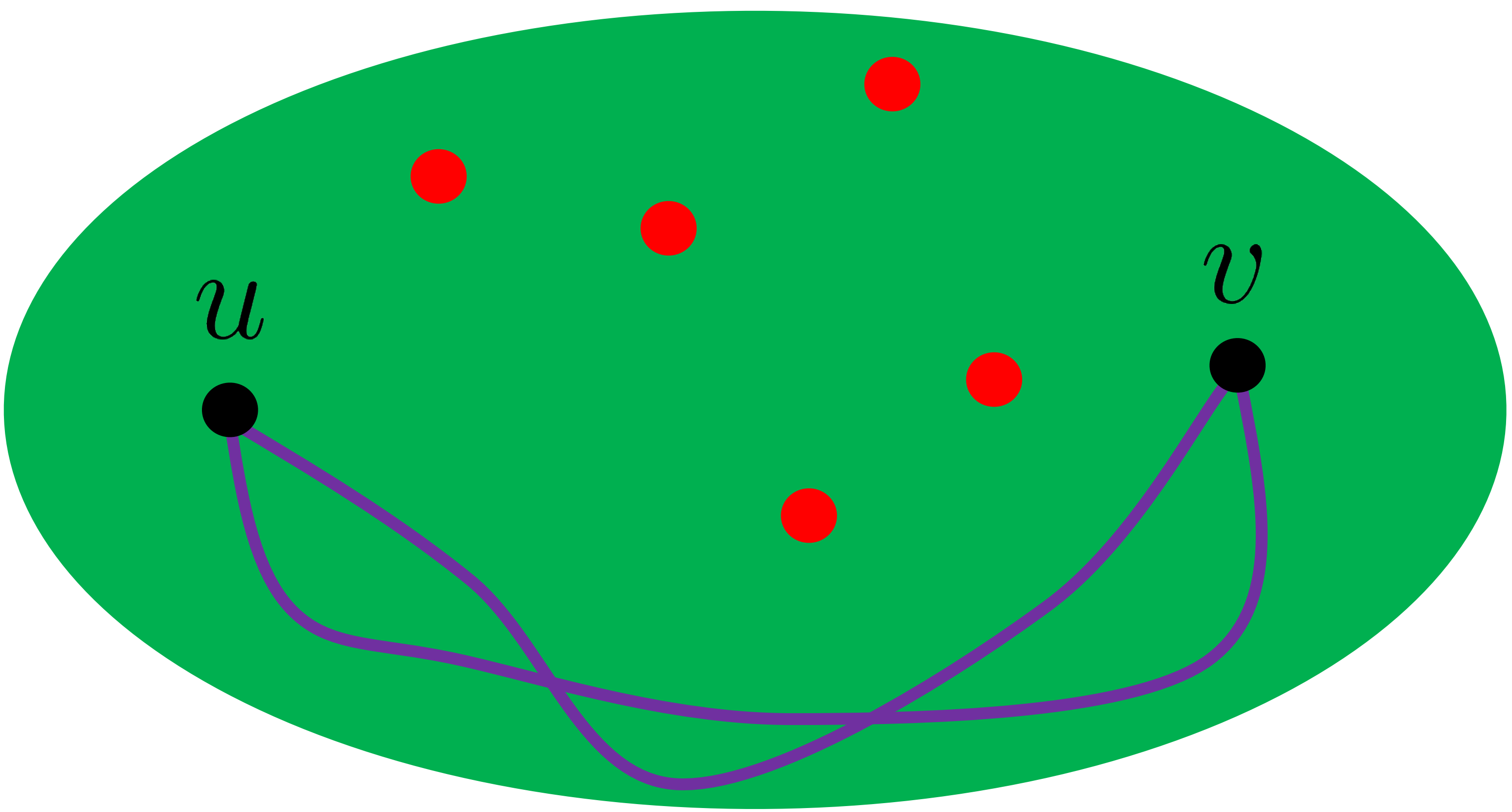
Counting shortest paths



Counting shortest paths

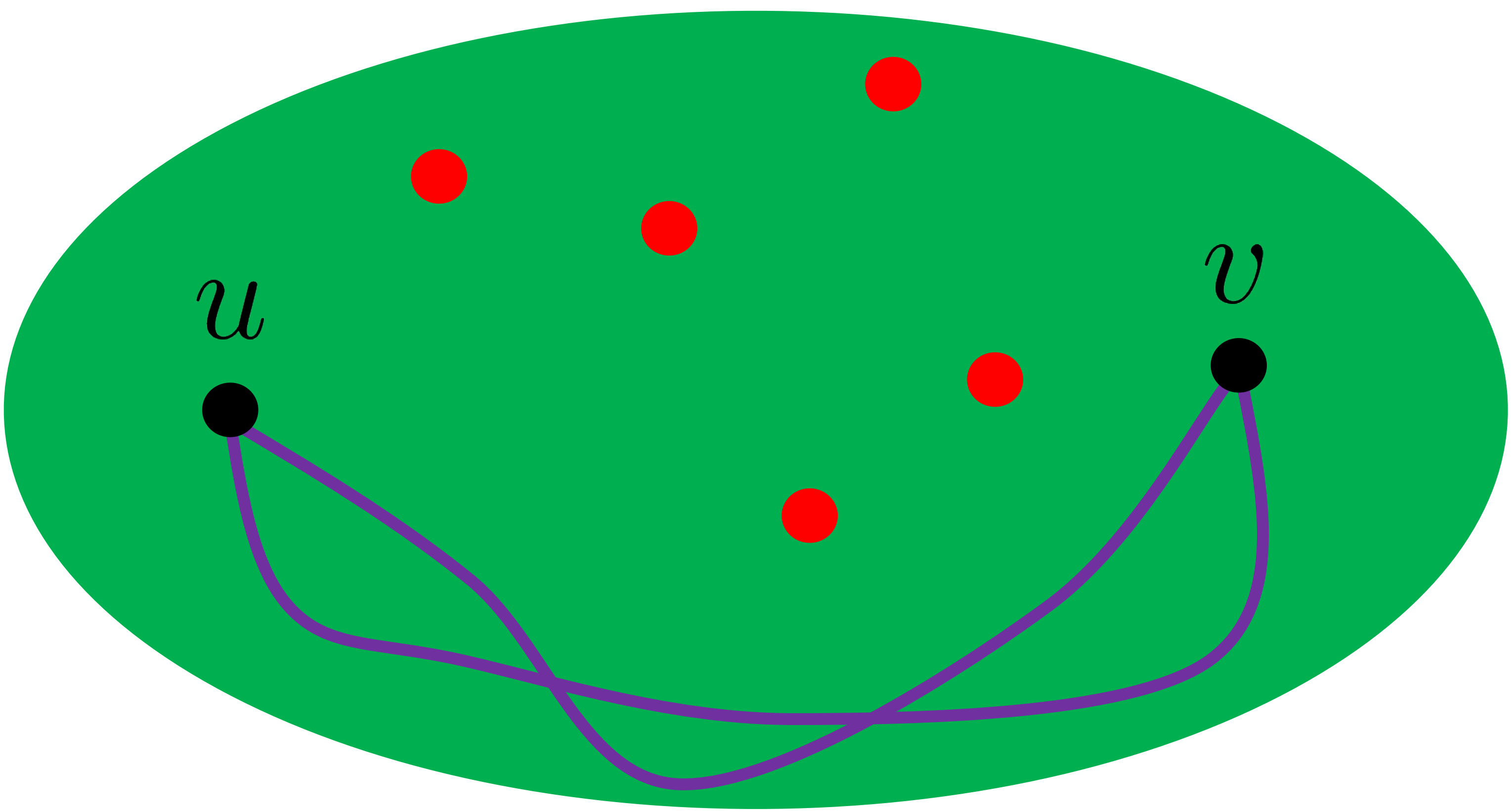


Counting shortest paths



Counting shortest paths

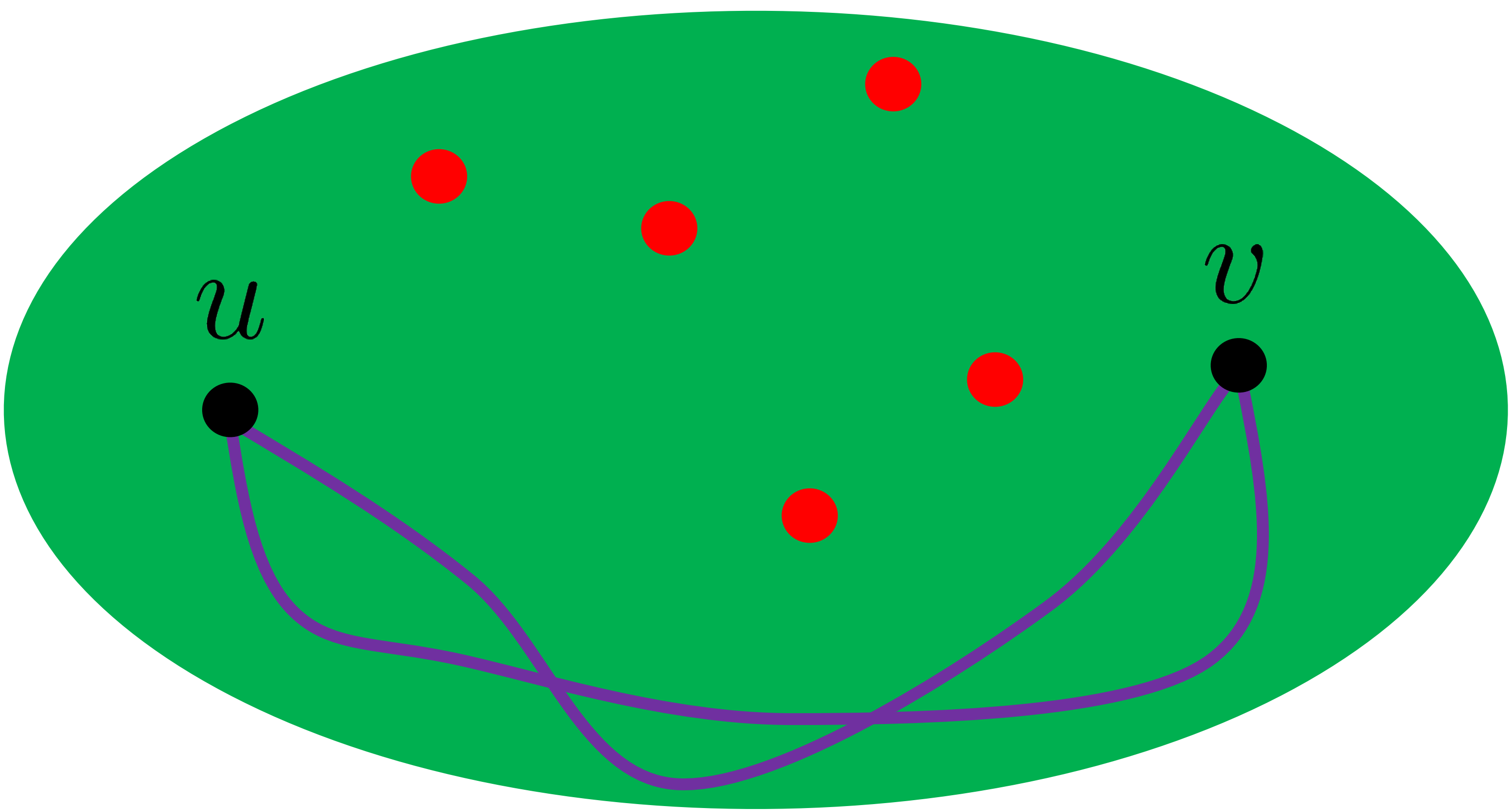
Classic $\tilde{O}(\sqrt{n})$ labels for counting can be used in fault tolerant setting



Counting shortest paths

Classic $\tilde{O}(\sqrt{n})$ labels for counting can be used in fault tolerant setting

Query time $\tilde{O}(\sqrt{n} \cdot k)$



Thanks for watching!!

Thanks for watching!!