

Reasoning about consistency choices in distributed systems

Hongseok Yang
University of Oxford

Joint work with Alexey Gotsman (IMDEA, Spain), Carla Ferreira (U Nova Lisboa), Mahsa Najafzadeh, Marc Shapiro (INRIA)

Global-scale Internet service

The screenshot shows the Amazon.com homepage. At the top, there's a navigation bar with the Amazon logo, a search bar, and links for 'Shop by Department', 'Your Amazon.com', 'Today's Deals', 'Gift Cards', 'Sell', 'Help', 'Hello. Sign in Your Account', 'Try Prime', and 'Wish List'. A banner for 'July 15th is Prime Day' is visible. Below the navigation, a large promotional banner for 'Try Prime Instant Video Free' is displayed, featuring a green play button icon and the text 'to stream thousands of movies & TV shows'. To the right of the text is a grid of movie and TV show covers, including titles like 'VEEP', 'NOAH', 'TRANSPARENT', 'Laggie', 'HANNIBAL', 'EMPIRE', 'ABBEY ROAD', 'SUN', 'CIRQUE NOIR', 'CAPTIVE', 'DEFIANCE', 'BOSCH', 'ELMO', 'HUNG', 'JACK BOY', 'SpongeBob SquarePants', 'THE WHITE QUEEN', 'STRANGE BLACK', 'UNDER THE DOME', and 'SHERLOCK'.

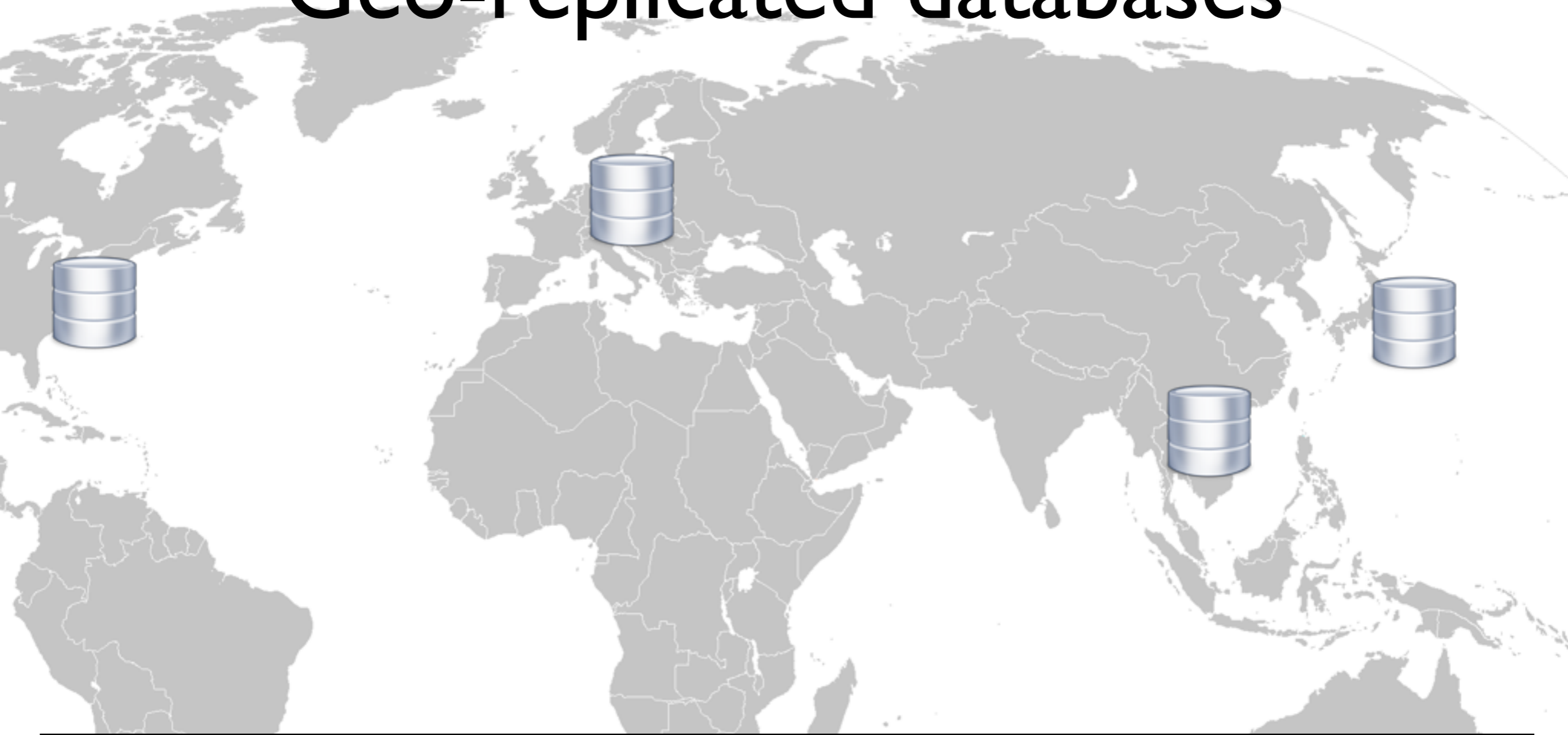
Movies Included with Prime Membership at No Additional Cost

[See more](#)



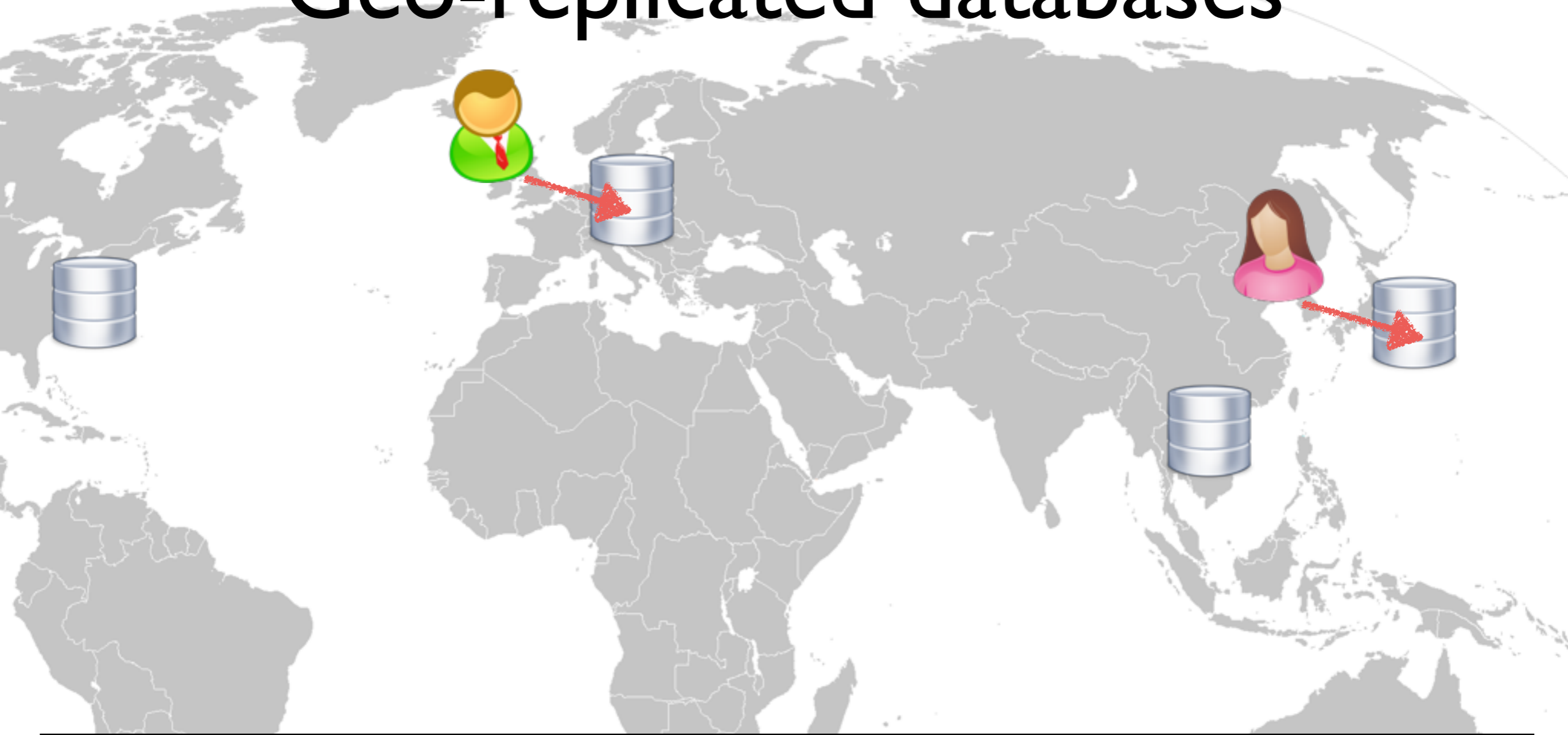
Instantly watch movies & TV shows
amazon Prime
Start Free Trial
Watch as much as you want
Anytime you want

Geo-replicated databases



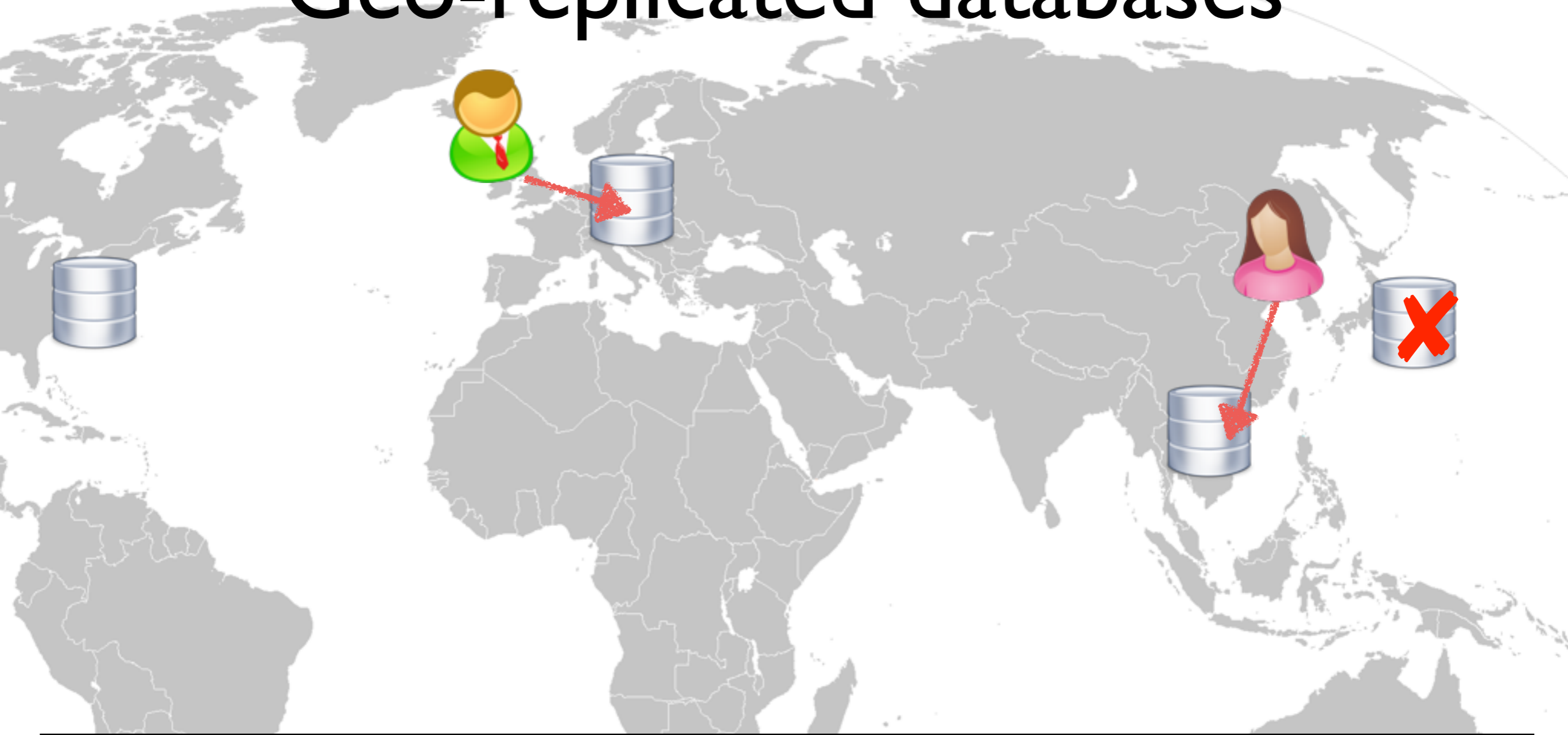
- Every data centre stores a complete replica of data
- Purpose: Minimising latency. Fault tolerance.

Geo-replicated databases



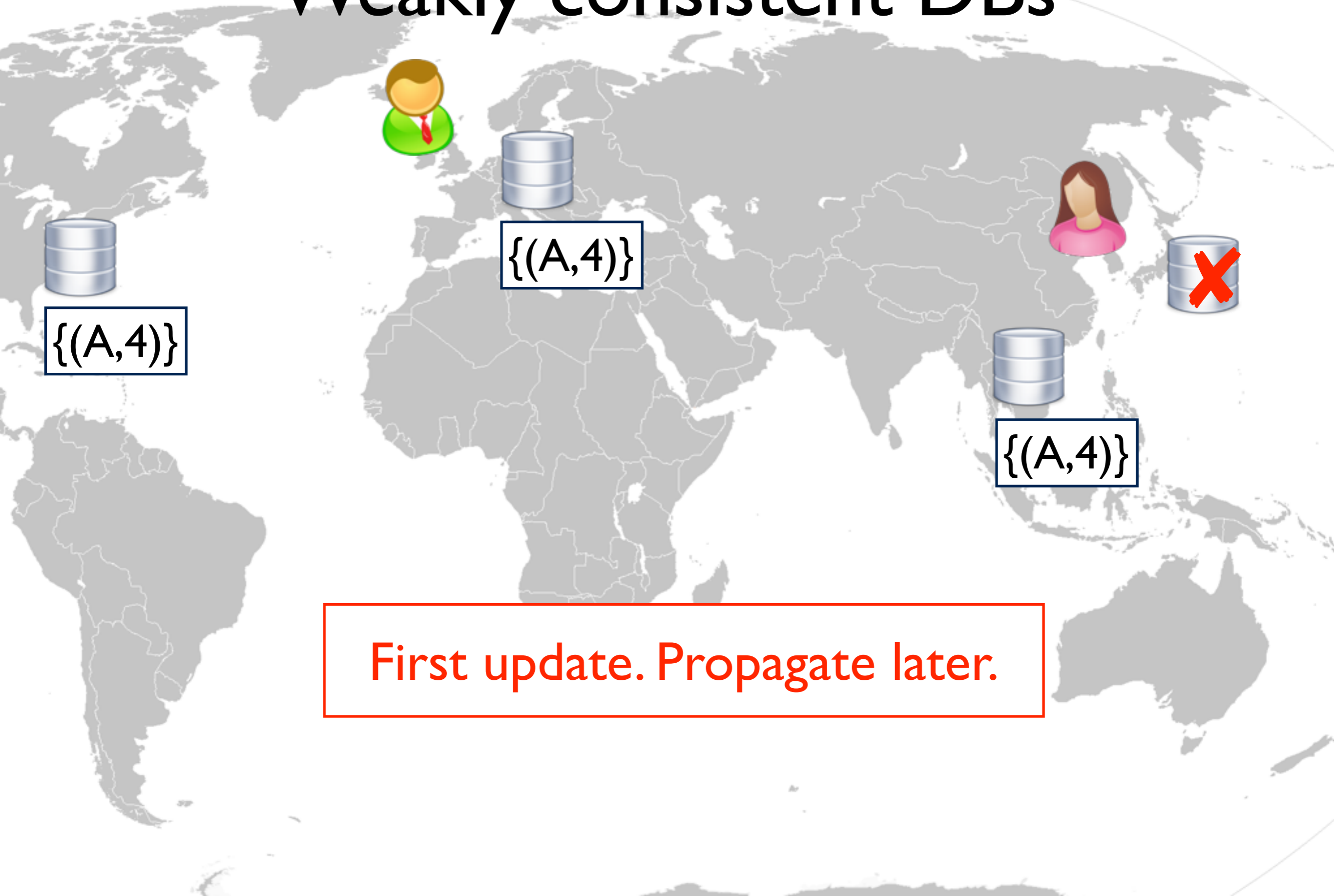
- Every data centre stores a complete replica of data
- Purpose: **Minimising latency.** Fault tolerance.

Geo-replicated databases



- Every data centre stores a complete replica of data
- Purpose: Minimising latency. **Fault tolerance.**

Weakly consistent DBs



$\{(A,4)\}$

$\{(A,4)\}$

$\{(A,4)\}$

First update. Propagate later.

Weakly consistent DBs



$\{(A,4)\}$



$\{(A,4)\}$

`cart.rem(A,2)`



$\{(A,2)\}$

First update. Propagate later.

Weakly consistent DBs



$\{(A,4)\}$



$\{(A,4)\}$

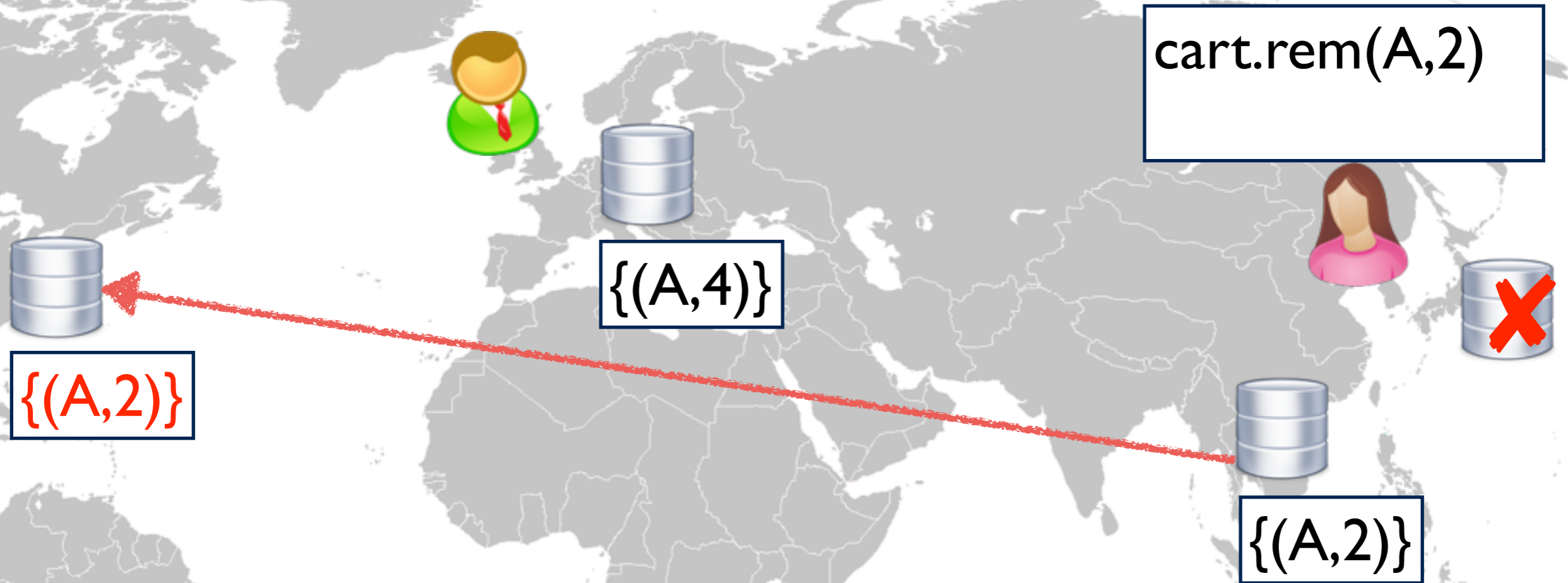
`cart.rem(A,2)`



$\{(A,2)\}$

First update. Propagate later.

Weakly consistent DBs



First update. Propagate later.

Weakly consistent DBs



$\{(A,2)\}$



$\{(A,4)\}$

`cart.rem(A,2)`
`cart.count(A): 4`

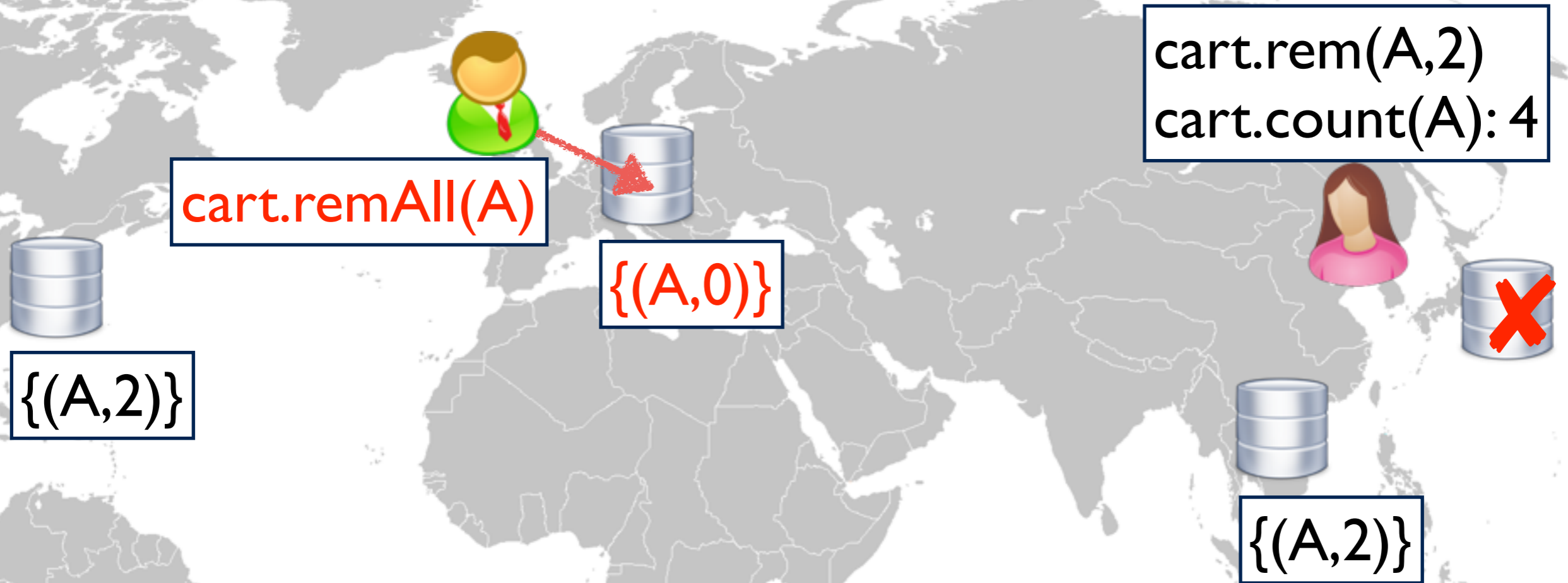


$\{(A,2)\}$

Issue 1: Anomalies

First update. Propagate later.

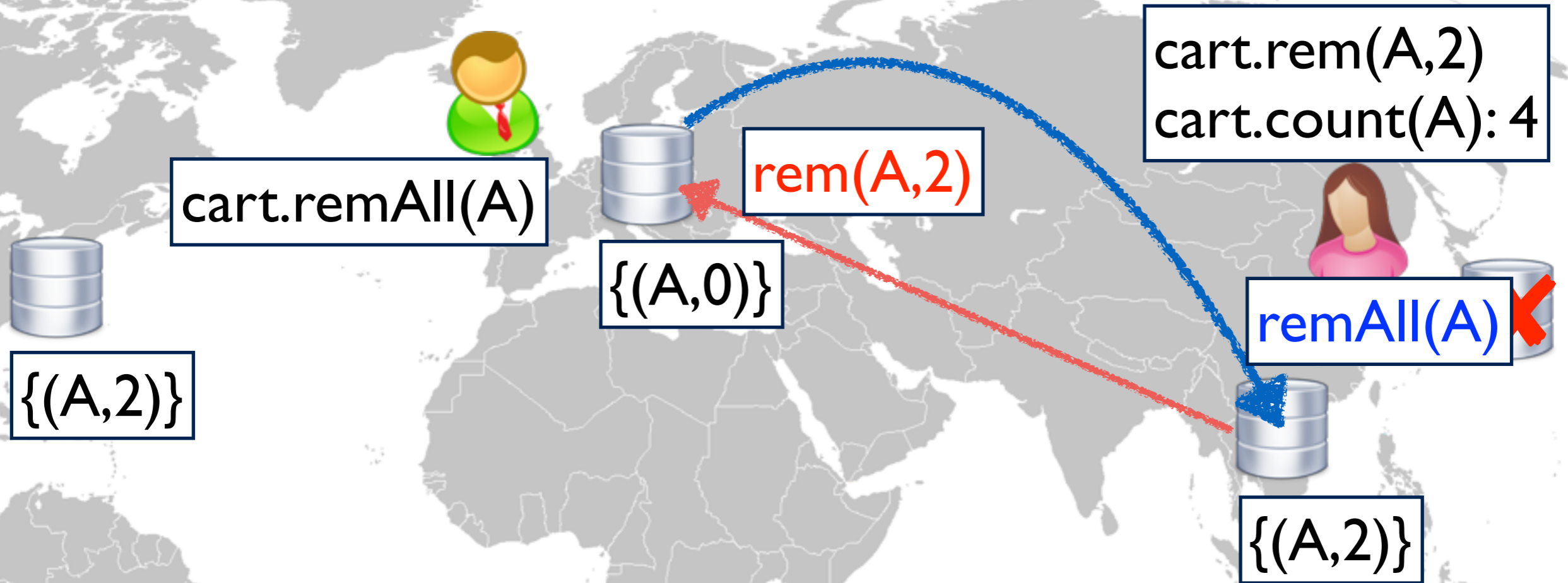
Weakly consistent DBs



Issue 2: Conflicting updates

First update. Propagate later.

Weakly consistent DBs



Issue 2: Conflicting updates

First update. Propagate later.

How to develop **correct** programs running on top of weakly consistent distributed databases?

How to develop **correct** programs running on top of weakly consistent distributed databases?

1. **Strengthen consistency selectively.**

How to develop **correct** programs running on top of weakly consistent distributed databases?

1. Strengthen consistency selectively.
2. **Prove the correctness of a program.**

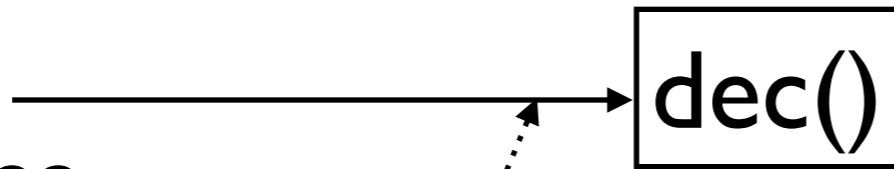
Simple bank account

```
class account {  
    // invariant: amount >= 0  
    var amount = 0  
  
    def query() = { return amount }  
  
    def inc() = {  
        amount = amount+1; return true  
    }  
  
    def dec() = {  
        if (amount > 0) {  
            amount = amount-1; return true  
        }  
        else { return false }  
    }  
}
```

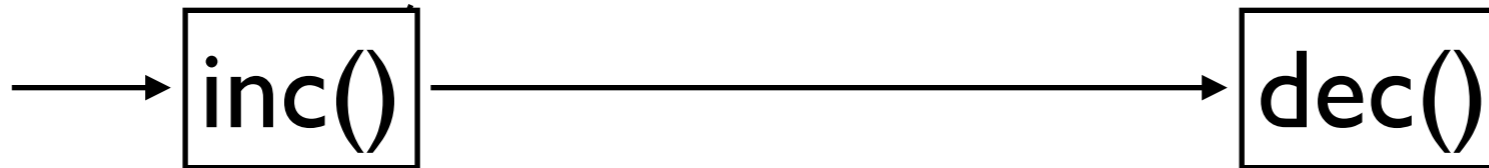
Distributed bank account

```
class account {  
  // invariant: amount >= 0  
  var[dis] amount = 0  
  
  def query() = { return (amount, (a)=>a) }  
  
  def inc() = {  
    amount = amount+1; return (true, (a)=>a+1)  
  }  
  
  def dec() = {  
    if (amount > 0) {  
      amount = amount-1; return (true, (a)=>a-1)  
    }  
    else { return (false, (a)=>a) }  
  }  
}
```

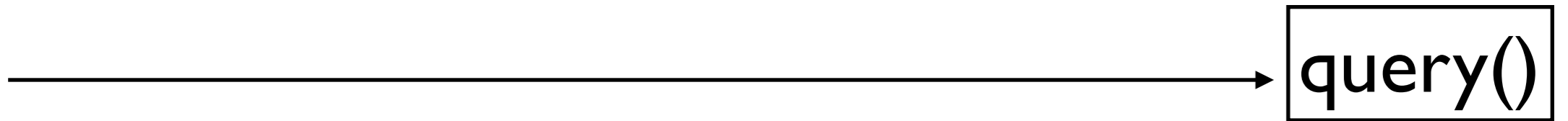
Alice
in Korea



Bob
in UK

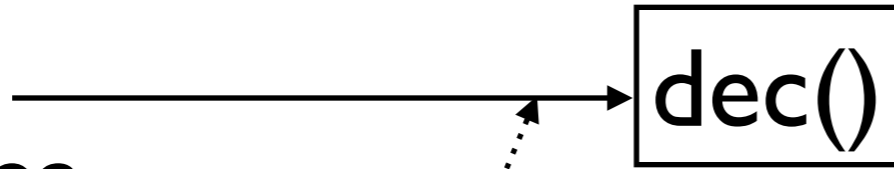


Carol
in USA

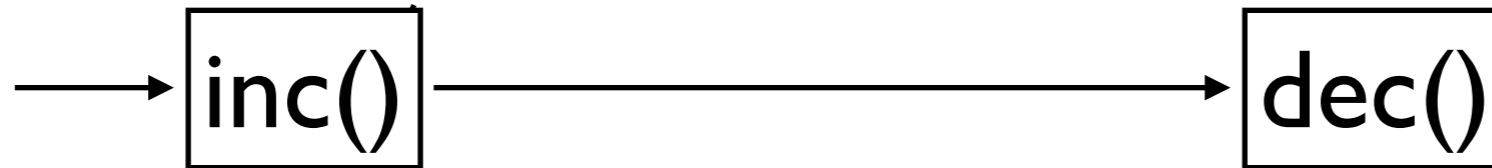


[Q] What cannot be the result of query()?
(a) 0 (b) -1 (c) -2 (d) all possible

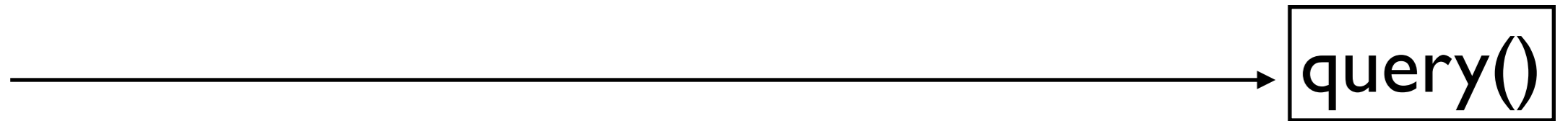
Alice
in Korea



Bob
in UK

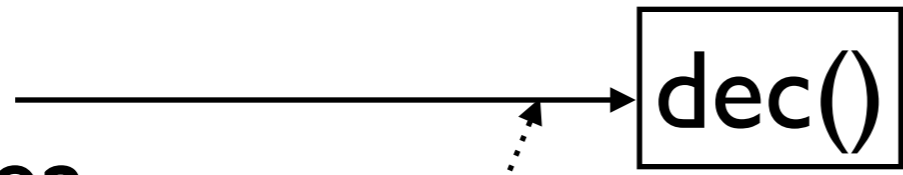


Carol
in USA

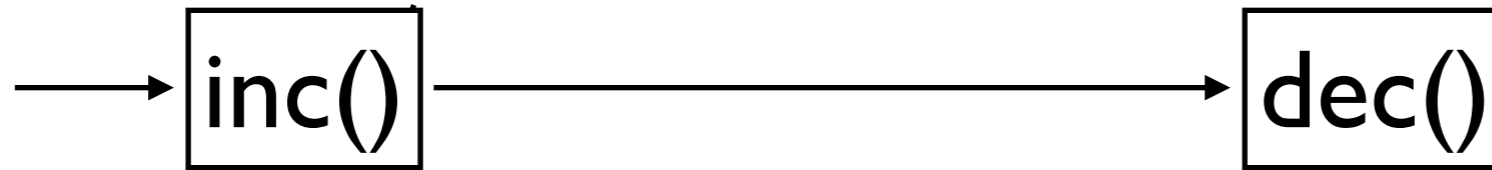


[Q] What cannot be the result of query()?
(a) 0 (b) -1 (c) -2 (d) all possible

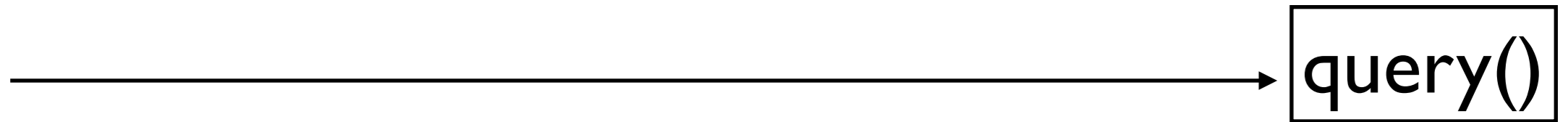
Alice
in Korea



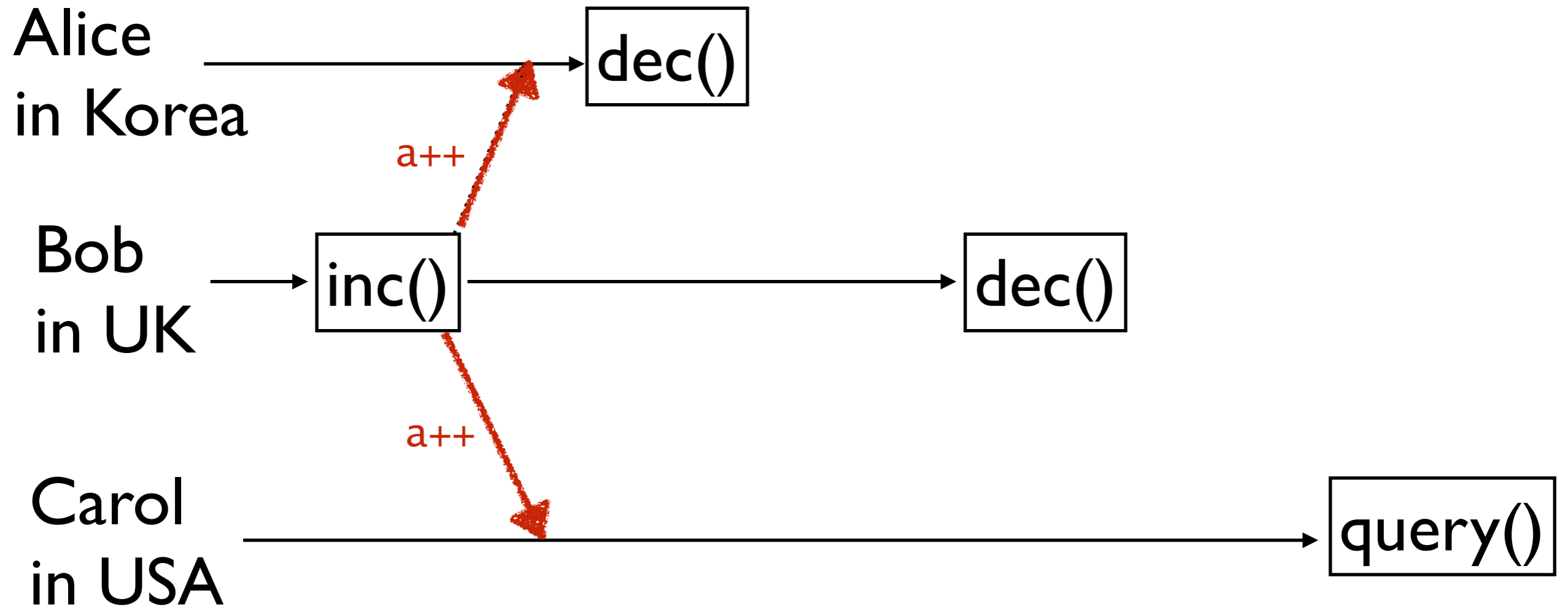
Bob
in UK



Carol
in USA

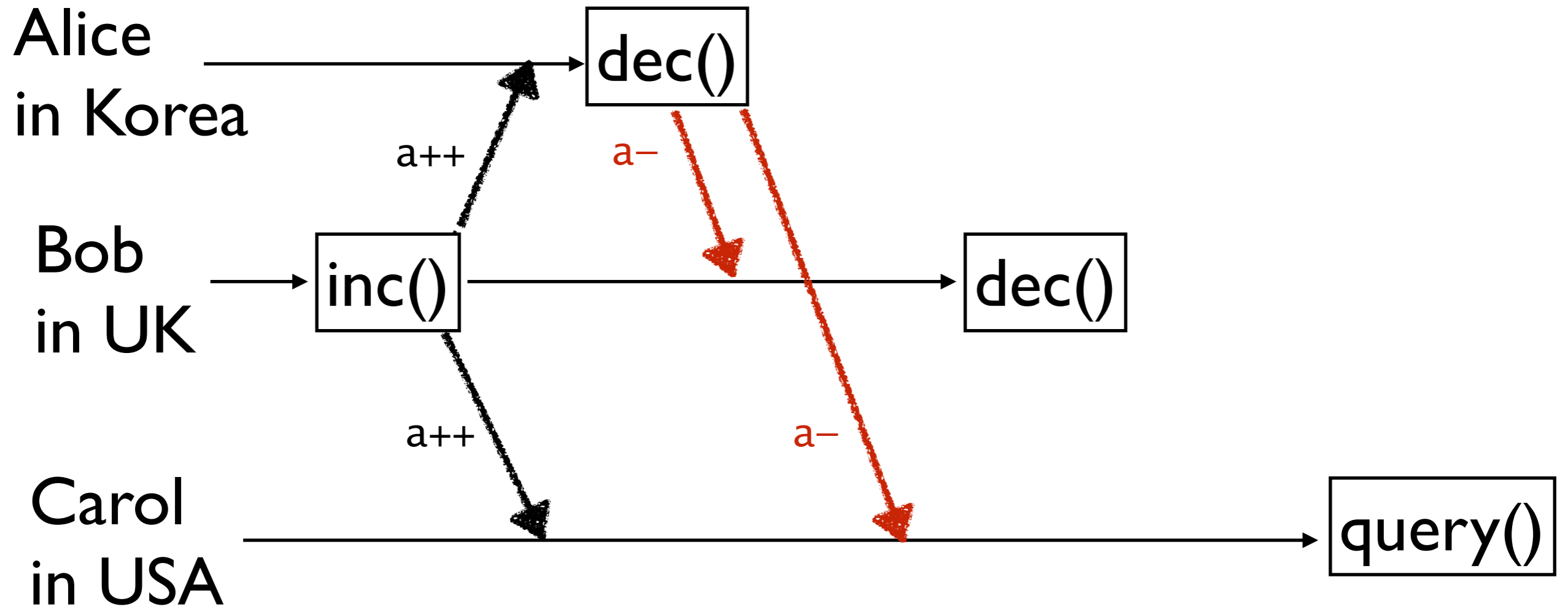


[Q] What cannot be the result of query()?
(a) 0 (b) -1 (c) -2 (d) all possible



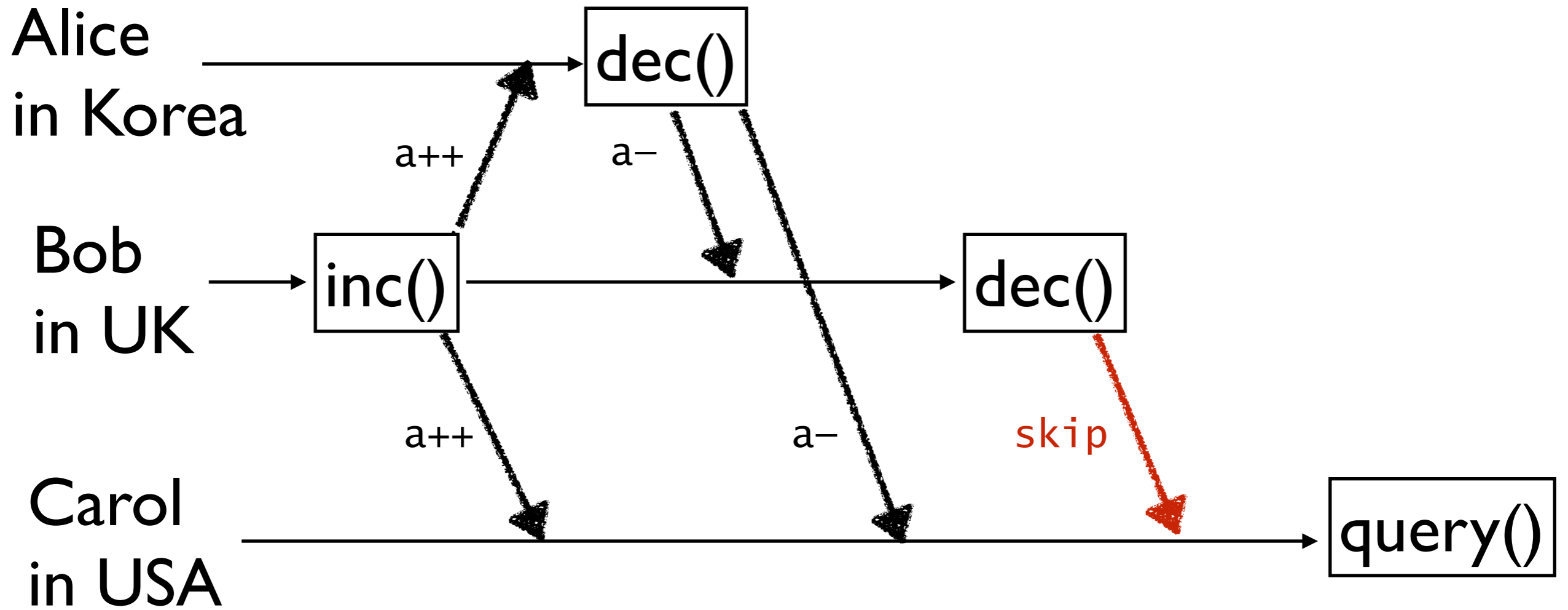
[Q] What cannot be the result of query()?

(a) 0 (b) -1 (c) -2 (d) all possible



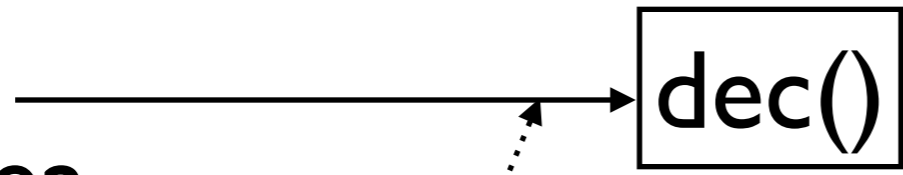
[Q] What cannot be the result of query()?

(a) 0 (b) -1 (c) -2 (d) all possible

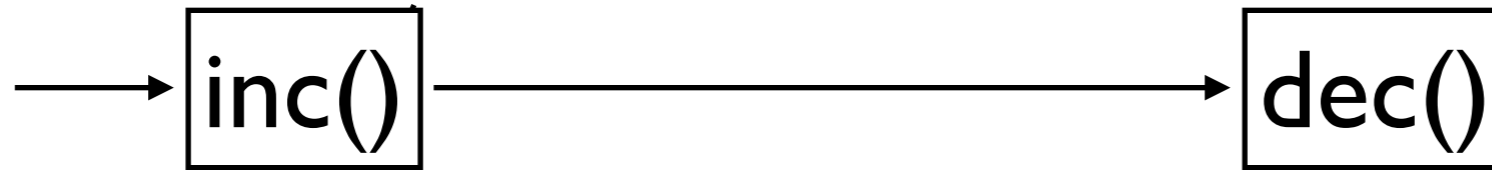


[Q] What cannot be the result of `query()`?
 (a) 0 (b) -1 (c) -2 (d) all possible

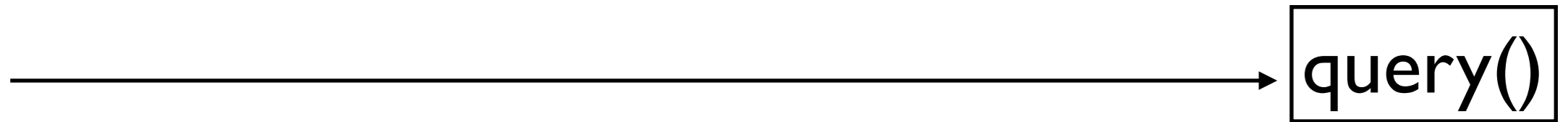
Alice
in Korea



Bob
in UK

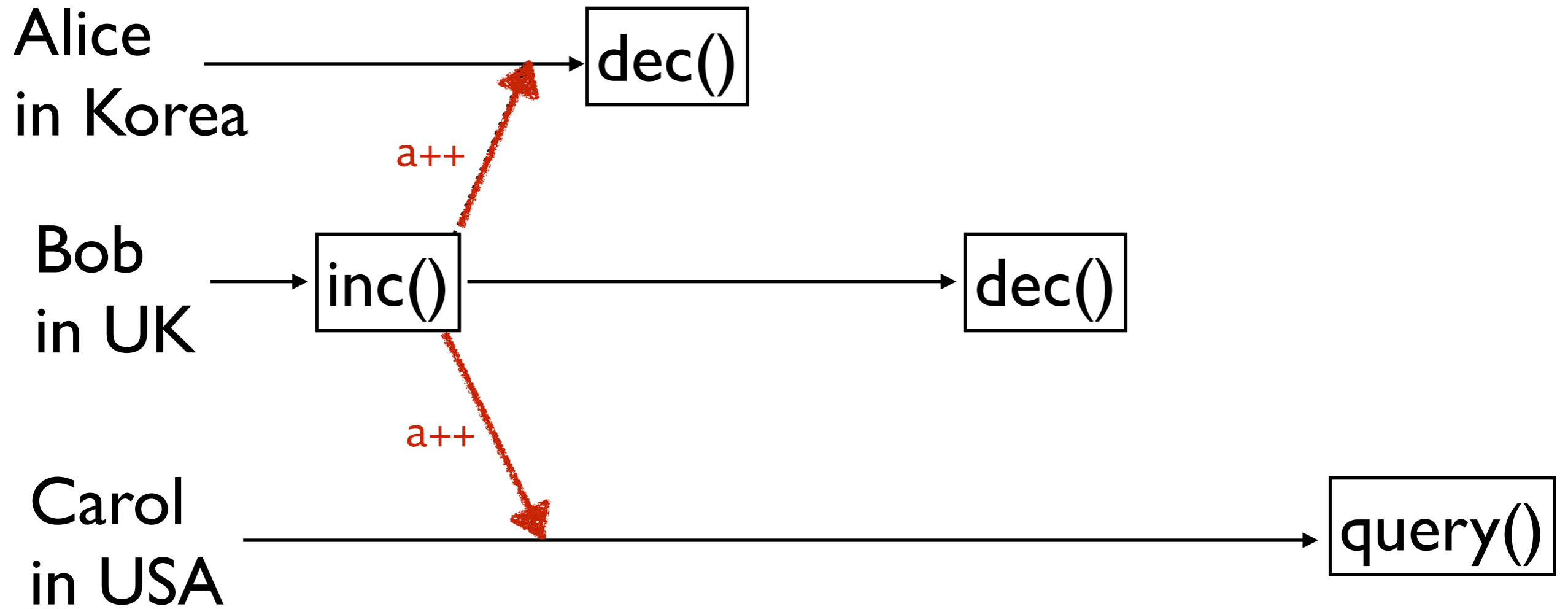


Carol
in USA



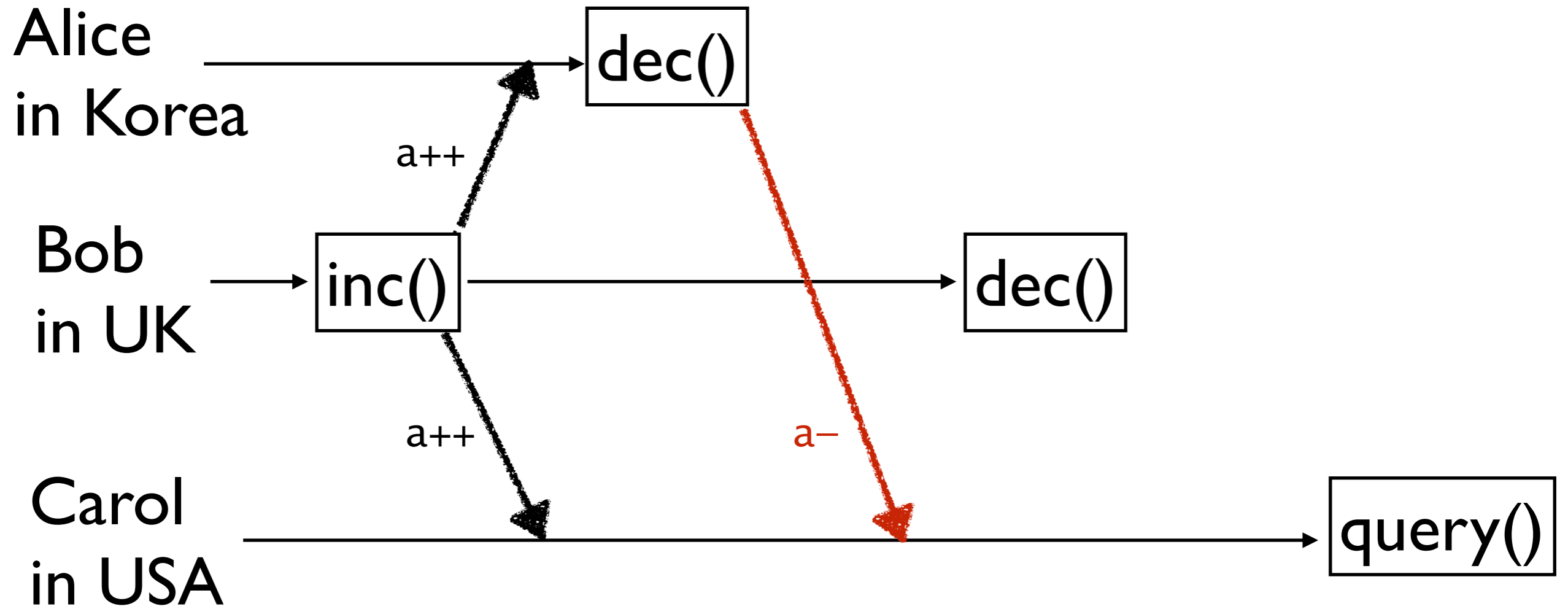
[Q] What cannot be the result of query()?

- (a) 0 (b) -1 (c) -2 (d) all possible



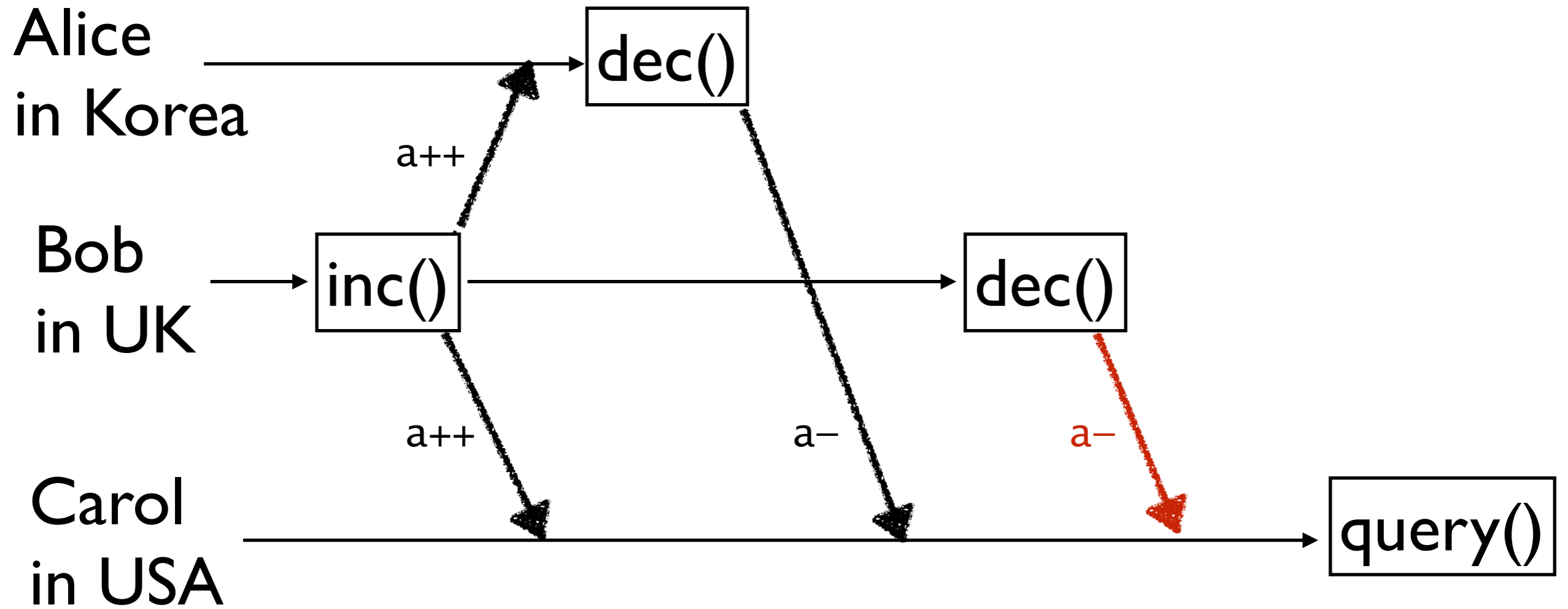
[Q] What cannot be the result of query()?

(a) 0 (b) -1 (c) -2 (d) all possible

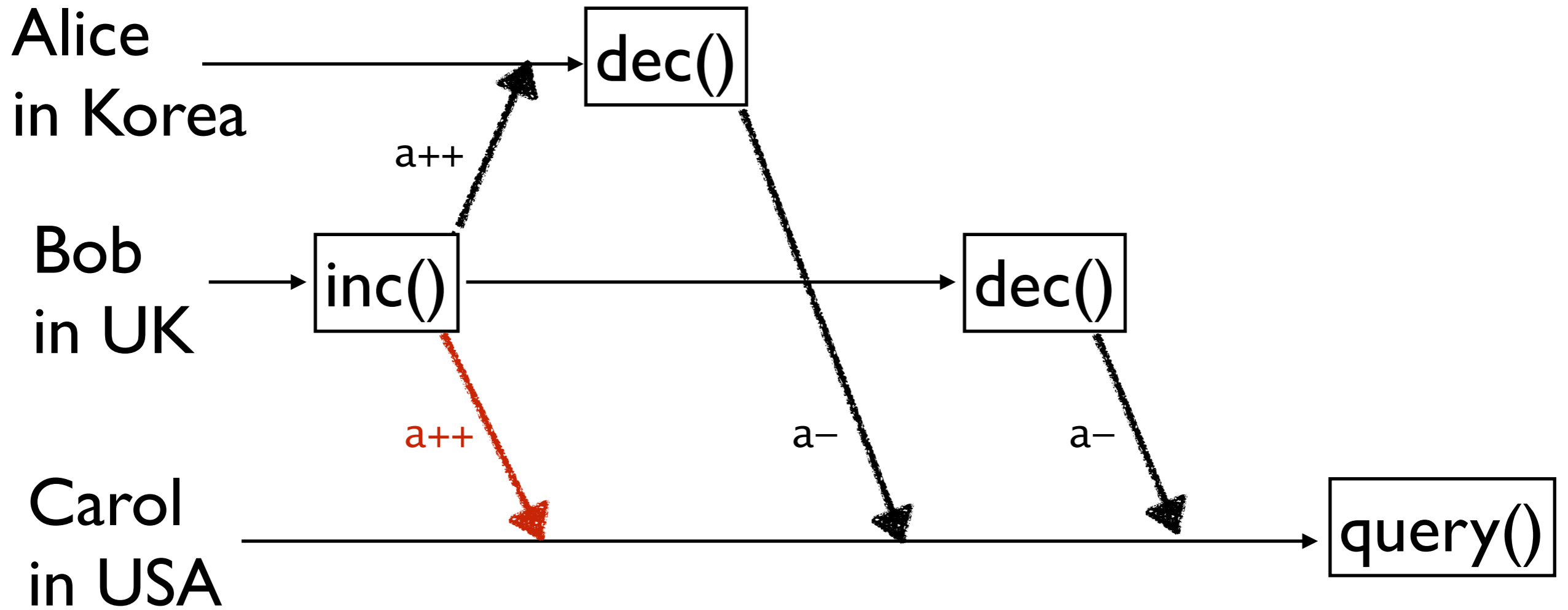


[Q] What cannot be the result of query()?

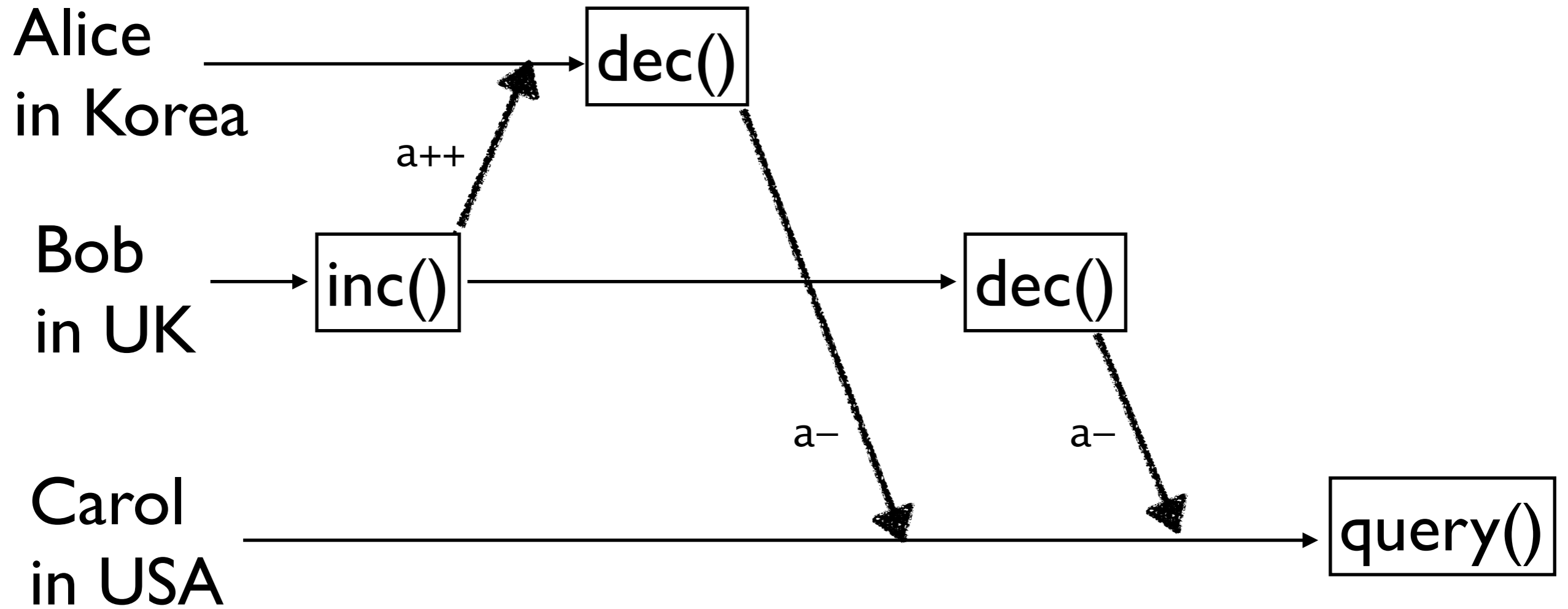
(a) 0 (b) -1 (c) -2 (d) all possible



[Q] What cannot be the result of `query()`?
 (a) 0 (b) -1 (c) -2 (d) all possible



[Q] What cannot be the result of `query()`?
 (a) 0 (b) -1 (c) -2 (d) all possible



[Q] What cannot be the result of `query()`?
(a) 0 (b) -1 (c) -2 (d) all possible

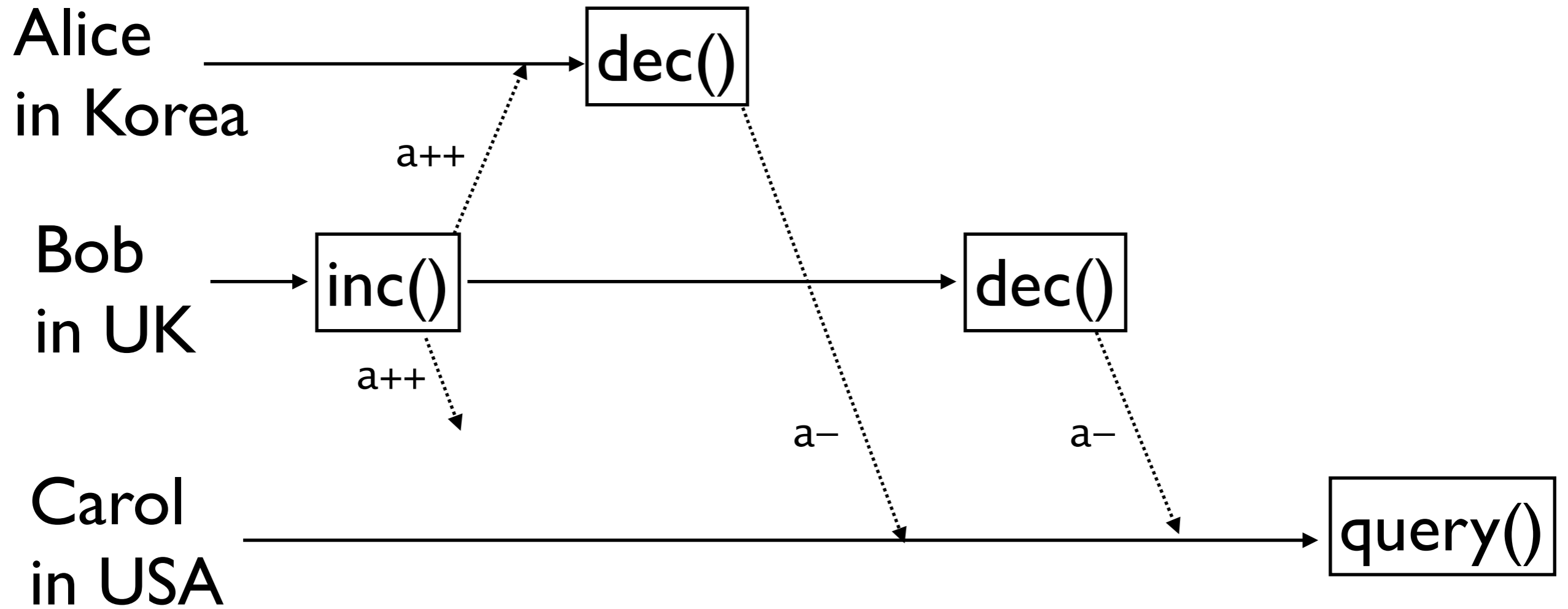
How to write correct prog.?

1. Strengthen consistency selectively.
2. Prove the correctness of a program.

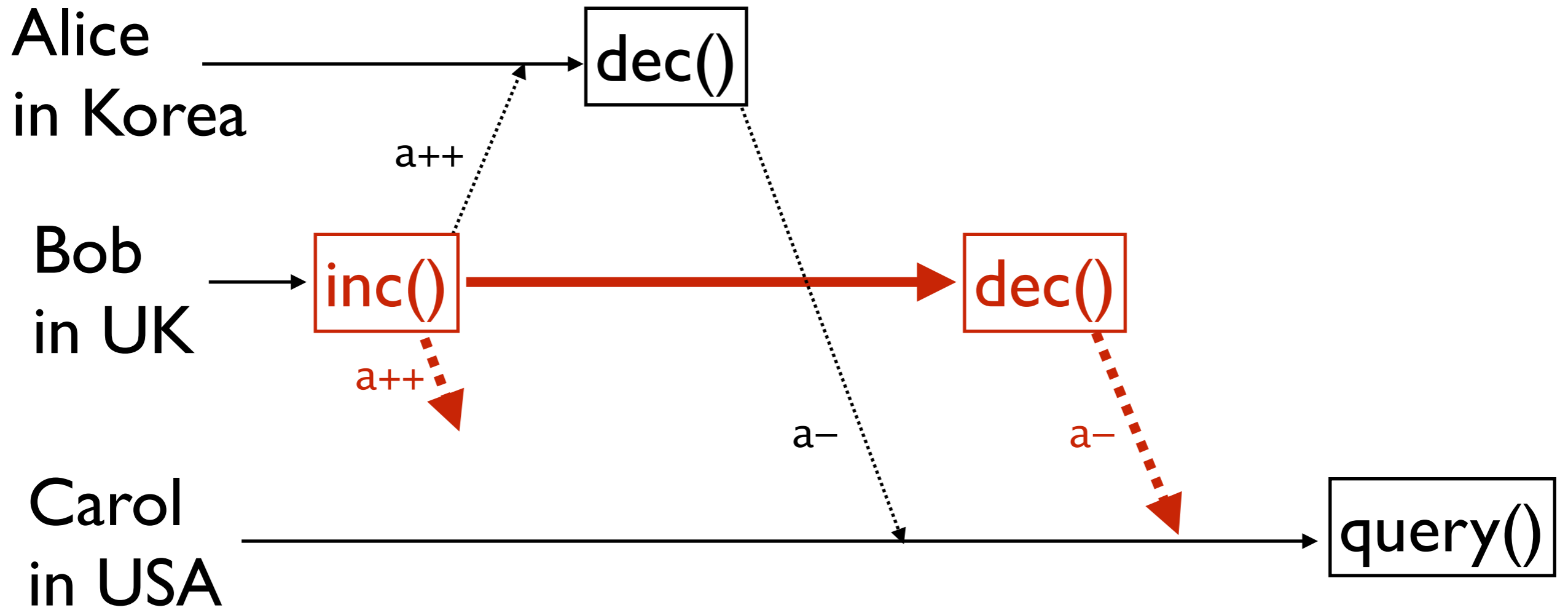
Causal consistency

- Message delivery preserves the dependency of events.

Axiom: HB is transitive.



[Q] What cannot be the result of query()?
(a) 0 (b) -1 (c) -2 (d) all possible



Not causally consistent.

[Q] What cannot be the result of query()?

(a) 0 (b) -1 (c) -2 (d) all possible

use causality

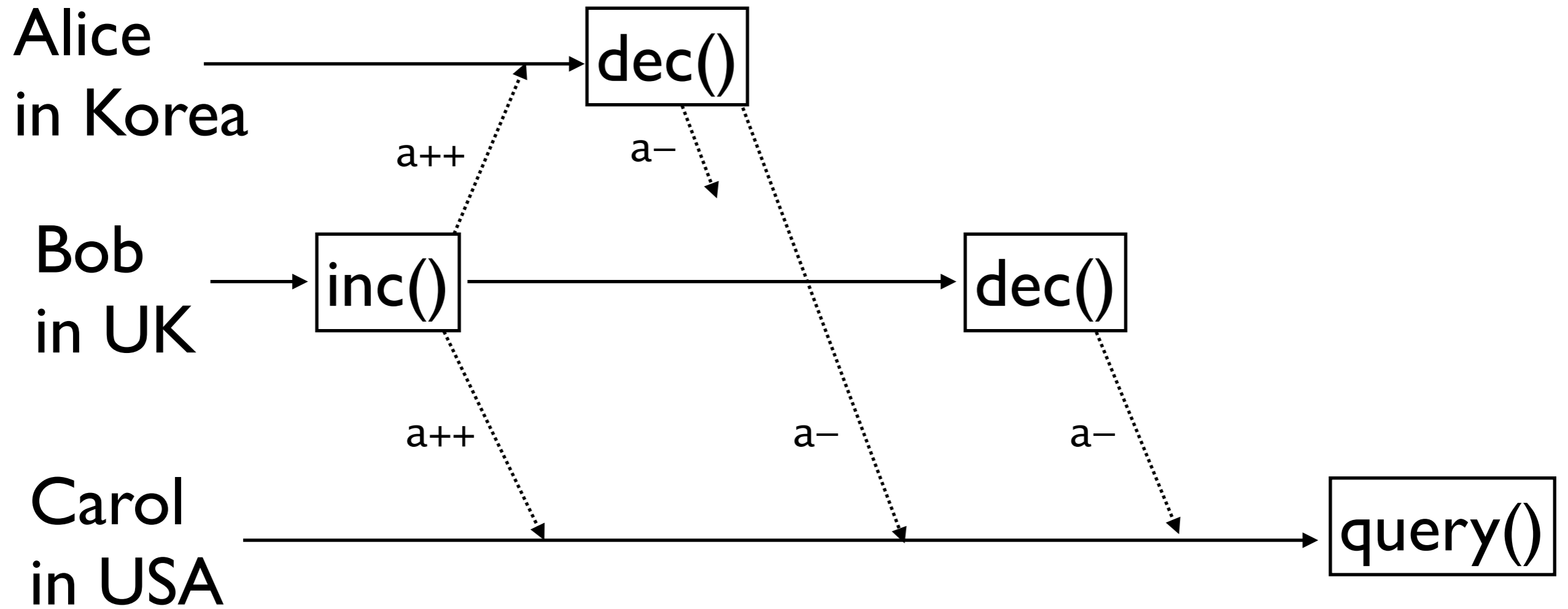
```
class account {  
  // invariant: amount >= 0  
  var[dis] amount = 0  
  
  def query() = { return (amount, (a)=>a) }  
  
  def inc() = {  
    amount = amount+1; return (true, (a)=>a+1)  
  }  
  
  def dec() = {  
    if (amount > 0) {  
      amount = amount-1; return (true, (a)=>a-1)  
    }  
    else { return (false, (a)=>a) }  
  }  
}
```

Token system

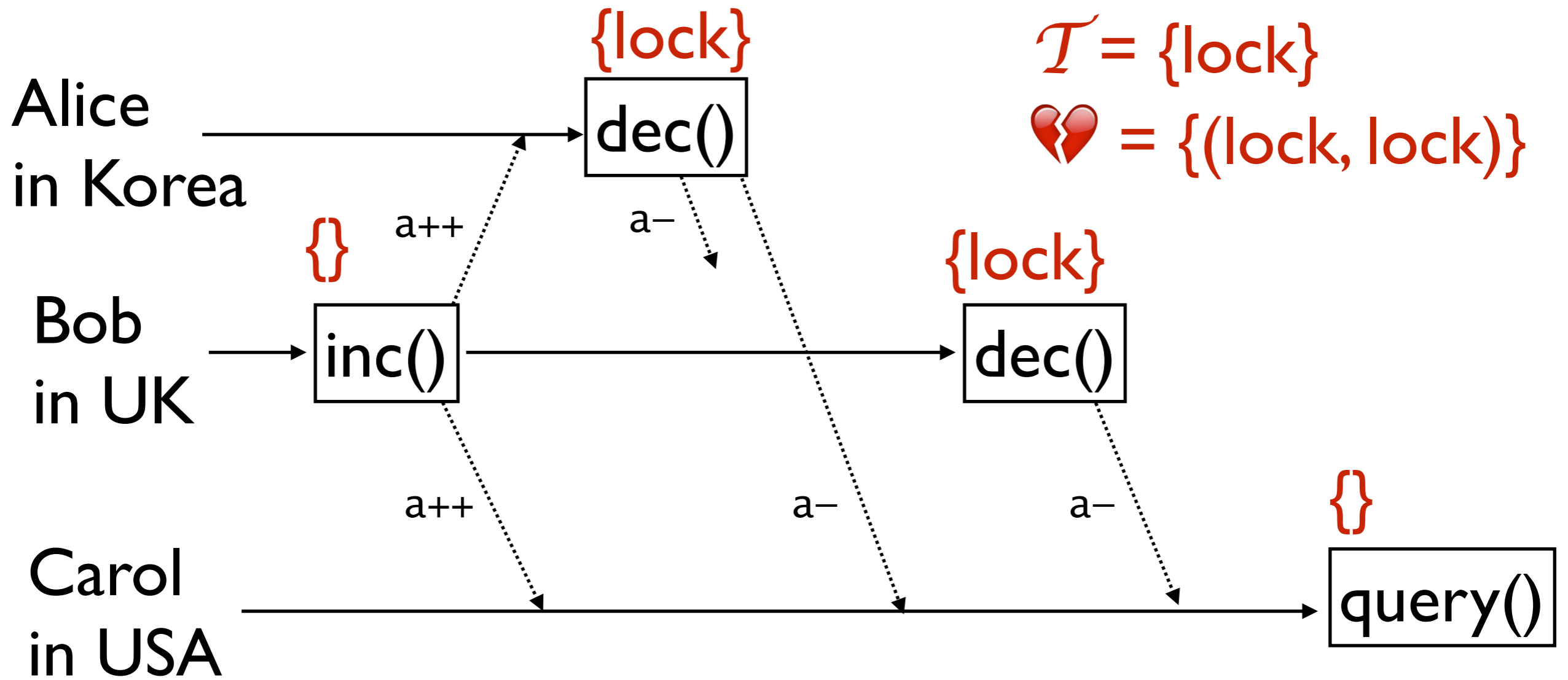
- $(\mathcal{I}, \heartsuit)$ where \heartsuit is a symmetric rel. on \mathcal{I} .
- Examples:
 1. $\mathcal{I} = \{\text{lock}\}$, $\heartsuit = \{(\text{lock}, \text{lock})\}$
 2. $\mathcal{I} = \{\text{rd}, \text{wr}\}$, $\heartsuit = \{(\text{rd}, \text{wr}), (\text{wr}, \text{wr}), (\text{wr}, \text{rd})\}$

On-demand consistency using a token system (\mathcal{T} ,)

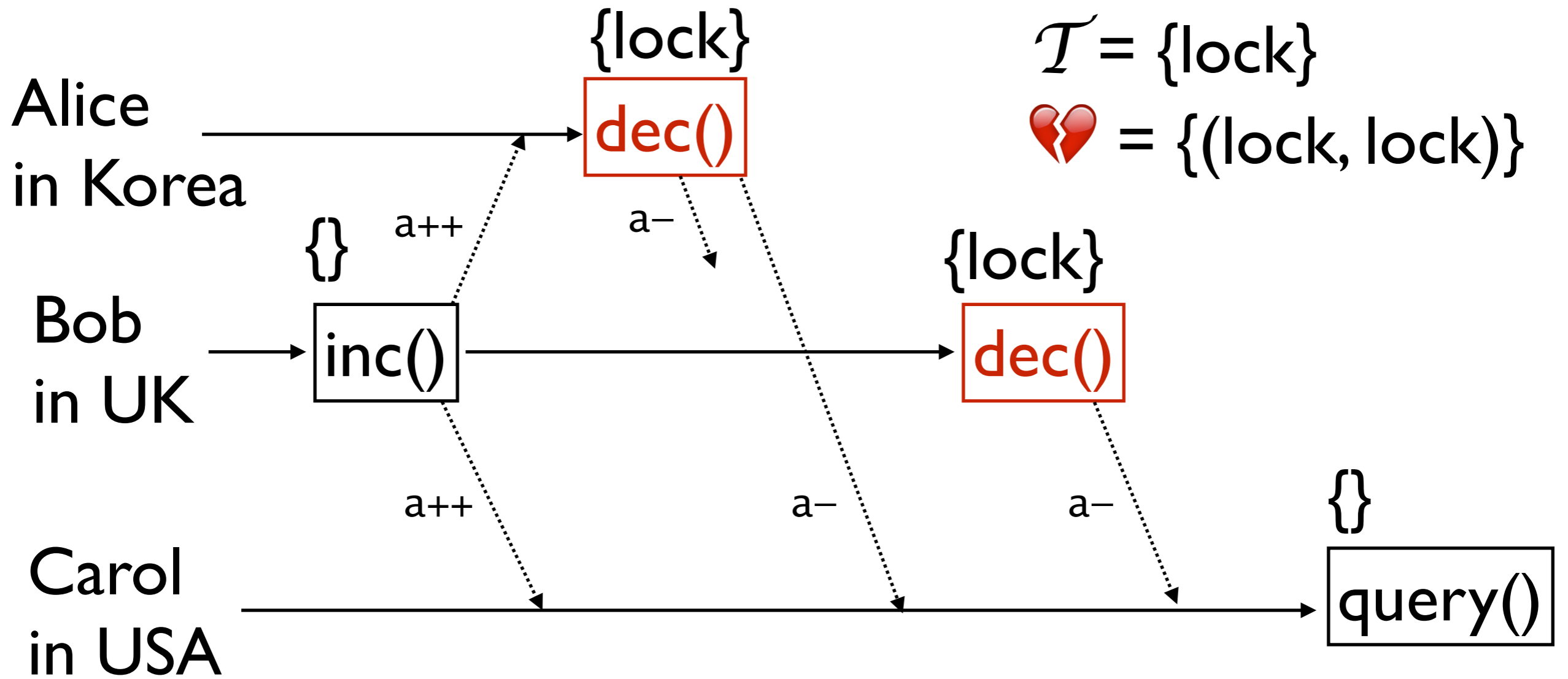
- Each operation acquires a set of tokens.
- Operations with conflicting tokens cannot be run concurrently.



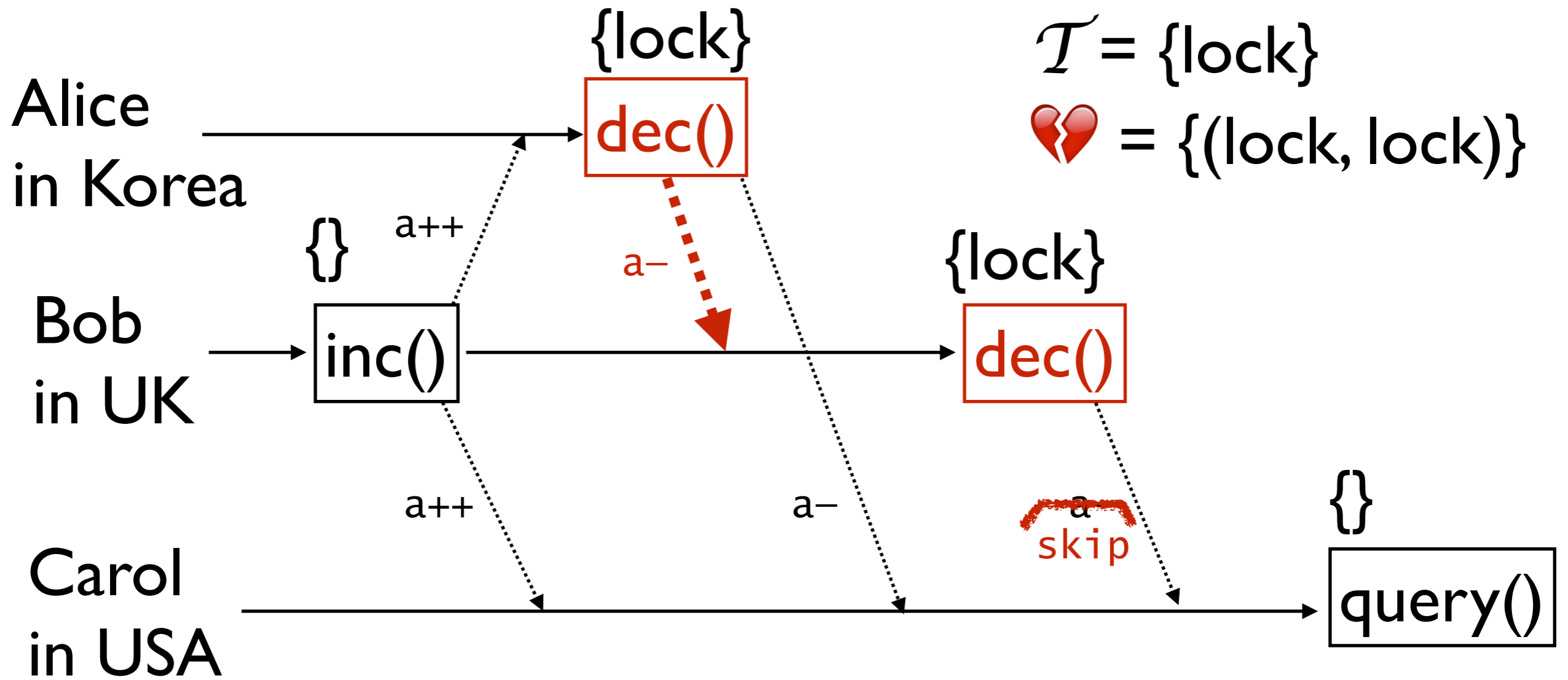
[Q] What cannot be the result of `query()`?
 (a) 0 (b) -1 (c) -2 (d) all possible



[Q] What cannot be the result of `query()`?
 (a) 0 (b) -1 (c) -2 (d) all possible



[Q] What cannot be the result of query()?
 (a) 0 **(b) -1** (c) -2 (d) all possible



[Q] What cannot be the result of query()?

- (a) 0 (b) -1 (c) -2 (d) all possible

use causality

```
class account {  
  // invariant: amount >= 0  
  var[dis] amount = 0  
  use-token-system({lock}, {(lock, lock)})  
  
  def query() with {} =  
  { return (amount, (a)=>a) }  
  
  def inc() with {} = {  
    amount = amount+1; return (true, (a)=>a+1)  
  }  
  
  def dec() with {lock} = {  
    if (amount > 0) {  
      amount = amount-1; return (true, (a)=>a-1)  
    }  
    else { return (false, (a)=>a) }  
  }  
}
```

How to write correct prog.?

1. Strengthen consistency selectively.
2. Prove the correctness of a program.

Our proof rule

- Based on rely-guarantee.
- Incorporates guarantees from causal and on-demand consistency.

To prove that J is an invariant

To prove that I is an invariant

$$\exists G_0 \in \mathcal{P}(\text{State} \times \text{State}), G \in \text{Token} \rightarrow \mathcal{P}(\text{State} \times \text{State})$$

To prove that I is an invariant

$\exists G_0 \in \mathcal{P}(\text{State} \times \text{State}), G \in \text{Token} \rightarrow \mathcal{P}(\text{State} \times \text{State})$

S1. $\sigma_{\text{init}} \in I$

To prove that I is an invariant

$\exists G_0 \in \mathcal{P}(\text{State} \times \text{State}), G \in \text{Token} \rightarrow \mathcal{P}(\text{State} \times \text{State})$

S1. $\sigma_{\text{init}} \in I$

S2. $G_0(I) \subseteq I \wedge \forall \tau. G(\tau)(I) \subseteq I$

To prove that I is an invariant

$\exists G_0 \in \mathcal{P}(\text{State} \times \text{State}), G \in \text{Token} \rightarrow \mathcal{P}(\text{State} \times \text{State})$

S1. $\sigma_{\text{init}} \in I$

S2. $G_0(I) \subseteq I \wedge \forall \tau. G(\tau)(I) \subseteq I$

S3. $\forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*)$
 $\implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))$

To prove that I is an invariant

$$\begin{aligned} \text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*) \\ \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma)) \end{aligned}$$

To prove that I is an invariant

$$\text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (\dots)^*) \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \dots$$

To prove that I is an invariant

$$\text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (\dots)^*) \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \dots$$

To prove that I is an invariant

$$\text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (\dots)^*) \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \dots$$

To prove that I is an invariant

$$\text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (\dots)^*) \\ \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \dots$$

To prove that I is an invariant

$$\text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (\dots)^*) \\ \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))$$

To prove that I is an invariant

$$\text{S3. } \forall \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*) \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))$$

$$\mathbb{T}^\perp = \{\tau \mid \nexists \tau' \in \mathbb{T}. (\tau, \tau') \in \heartsuit\}$$

$$I = \{ \sigma \mid 0 \leq \sigma \}$$

$$G_0 = \{ (\sigma, \sigma') \mid \sigma \leq \sigma' \}$$

$$G_1(\text{lock}) = \{ (\sigma, \sigma') \mid 0 \leq \sigma' \leq \sigma \}$$

$$\text{S3. } \forall \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G((\mathcal{F}_\sigma^{\text{tok}})^\perp))^*) \implies (\sigma', \mathcal{F}_\sigma^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_\sigma^{\text{tok}})$$

$$I = \{ \sigma \mid 0 \leq \sigma \}$$

$$G_0 = \{ (\sigma, \sigma') \mid \sigma \leq \sigma' \}$$

$$G_1(\text{lock}) = \{ (\sigma, \sigma') \mid 0 \leq \sigma' \leq \sigma \}$$

S3. $\forall \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*)$
dec() $\implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))$

$$I = \{ \sigma \mid 0 \leq \sigma \}$$

$$G_0 = \{ (\sigma, \sigma') \mid \sigma \leq \sigma' \}$$

$$G_1(\text{lock}) = \{ (\sigma, \sigma') \mid 0 \leq \sigma' \leq \sigma \}$$

$$\text{S3. } \forall \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G(\overbrace{(\mathcal{F}_o^{\text{tok}}(\sigma))^\perp}^{\{\}}))^*) \\ \text{dec()} \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\underbrace{\mathcal{F}_o^{\text{tok}}(\sigma)}_{\{\text{lock}\}})$$

$$I = \{ \sigma \mid 0 \leq \sigma \}$$

$$G_0 = \{ (\sigma, \sigma') \mid \sigma \leq \sigma' \}$$

$$G_1(\text{lock}) = \{ (\sigma, \sigma') \mid 0 \leq \sigma' \leq \sigma \}$$

G_0^*

$$S3. \forall \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in \{ \} (G_0 \cup G((\mathcal{F}_\sigma^{\text{tok}}(\sigma))^\perp))^*)$$

$$\text{dec}() \implies (\sigma', \mathcal{F}_\sigma^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_\sigma^{\text{tok}}(\sigma)) \{ \text{lock} \}$$

$$G_0 \cup G_1(\text{lock})$$

$$I = \{ \sigma \mid 0 \leq \sigma \}$$

$$G_0 = \{ (\sigma, \sigma') \mid \sigma \leq \sigma' \}$$

$$G_1(\text{lock}) = \{ (\sigma, \sigma') \mid 0 \leq \sigma' \leq \sigma \}$$

S3. $\forall \underline{0}, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in \boxed{G_0^*}) \wedge \text{dec}() \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \boxed{G_0 \cup G_1(\text{lock})}$

$$I = \{ \sigma \mid 0 \leq \sigma \}$$

$$G_0 = \{ (\sigma, \sigma') \mid \sigma \leq \sigma' \}$$

$$G_1(\text{lock}) = \{ (\sigma, \sigma') \mid 0 \leq \sigma' \leq \sigma \}$$

S3. $\forall \underline{0}, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in \boxed{G_0^*})$
 $\text{dec}() \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \boxed{G_0 \cup G_1(\text{lock})}$

if $0 < \sigma$ then $\sigma' - 1$ else σ'

What if no on-demand consistency?

$$\text{S3. } \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in (G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*) \\ \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))$$

What if no on-demand consistency?

$$\begin{aligned} S3. \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in \underbrace{(G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*}_{G^*}) \\ \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \underbrace{G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))}_{G} \end{aligned}$$

What if no causality?

$$\begin{aligned} S3. \forall o, \sigma, \sigma'. (\sigma \in I \wedge (\sigma, \sigma') \in \underbrace{(G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma))^\perp))^*}_{\mathbf{G}^*}) \\ \implies (\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in \underbrace{G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))}_{\mathbf{G}} \end{aligned}$$

What if no causality?

$\sigma' \in I$

S3. $\forall o, \sigma, \sigma'. (\sigma \in I \wedge \cancel{(o, \sigma') \in (G_0 \cup G((\mathcal{F}_o^{\text{tok}}(\sigma)) \perp))})^*$

$\implies \cancel{(\sigma', \mathcal{F}_o^{\text{eff}}(\sigma)(\sigma')) \in G_0 \cup G(\mathcal{F}_o^{\text{tok}}(\sigma))}$

$\mathcal{F}_o^{\text{eff}}(\sigma)(\sigma') \in I$