Database – Info Storage and Retrieval

- **Aim: Understand basics of**
  - Info storage and Retrieval;
  - Database Organization;
  - DBMS, Query and Query Processing;
  - Work some simple exercises;
  - Concurrency Issues (in Database)
- **Readings:**
  - [SG] --- Ch 13.3
- **Optional:**
  - Some experiences with MySQL, Access

Concurrency Issues

- **Concurrency**
  - **When 2 processes access *S* at the *same* time!**
  - **Can cause all kinds of problems**
  - **Important issue in *all* areas**
- **Illustrate with concurrency issue in database**
- **Readings for Concurrency Issues**
  - **Record Locking – Wikipedia**
    - http://en.wikipedia.org/wiki/Record_locking
  - **Read [SG3] Section 6.4.1 (pp. 268--272)**
    - Efficient Allocation and Safe Use of Resources

… from a true story…

- Close to CNY some years ago,
  - TAK went to POSB ATM machine
    Puts in ATM Card,
    Punches in PIN,
    Select withdrawal Type and Amount $200
  - ATM says… Please wait. Checking,
    Checking…
    Checking…
    Eventually, Machine times out,
    ATM card came back out,
    BUT NO $200 cash!

… from a true story… (2)

- **What did TAK do?**
  - **Looks around puzzled.**
  - **. . .**
- **Eventually, he tries again**
  - **Second time, he got lucky,**
  - **He got his $200 cash. He is HAPPY!**
- **But wait . . .**

WHAT DO YOU THINK HAPPENED?

Concurrency Issue: Simple Example

- Bank account info:

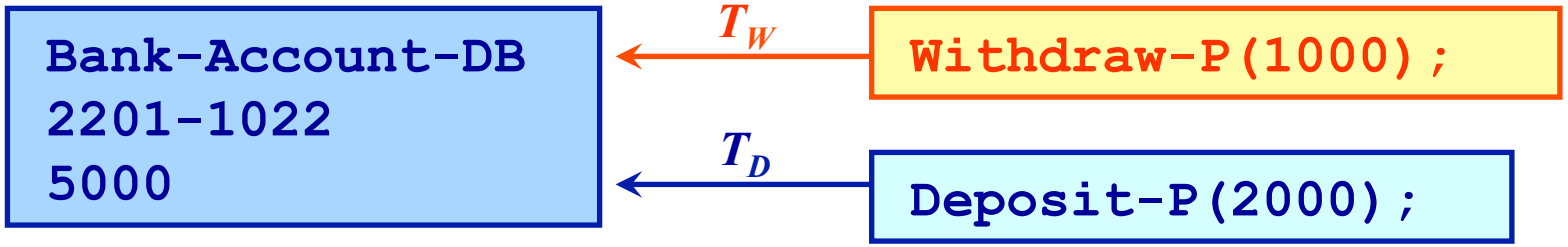
BANK-ACCOUNT-DB			
Account #	Name	Balance	Other Info…
2201-1022	Albert Bank	5000	…

- Deposit and Withdrawal Processes

```
Withdraw-P (amt);
begin
   check balance;
   bal ← bal - amt;
end;
```

```
Deposit-P (amt);
begin
   check balance;
   bal ← bal + amt;
end;
```

Normal Operations (1/2)

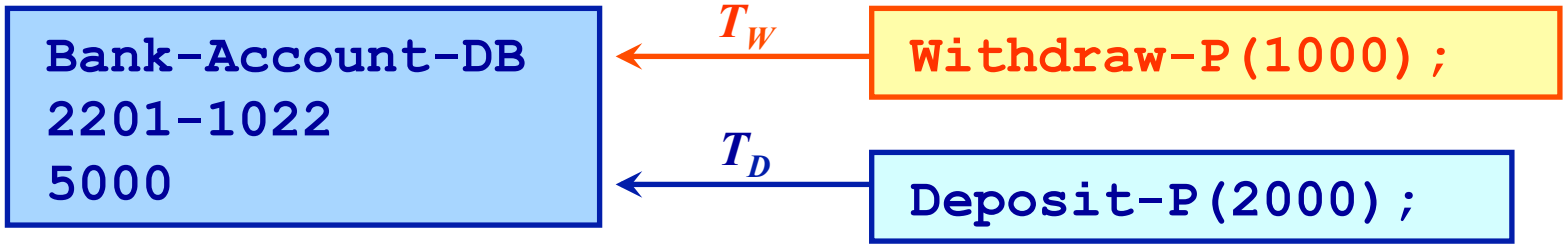


If $(T_W < T_D)$, then

Time	Withdraw-P(1000)	Deposit(2000)	Balance
T_W	Check balance [5000]		5000
T_W+1	Bal ← Bal – 1000		4000
. . .	. . .	. . .	4000
T_D		Check balance [4000]	4000
T_D+1		Bal ← Bal + 2000	6000

Correct balance

Normal Operations (2/2)



If $(T_W > T_D)$, then

Time	Withdraw-P(1000)	Deposit(2000)	Balance
T_D		Check balance [5000]	5000
T_D+1		Bal ← Bal + 2000	7000
. . .	. . .	. . .	7000
T_W	Check balance [7000]		7000
T_W+1	Bal ← Bal – 1000		6000

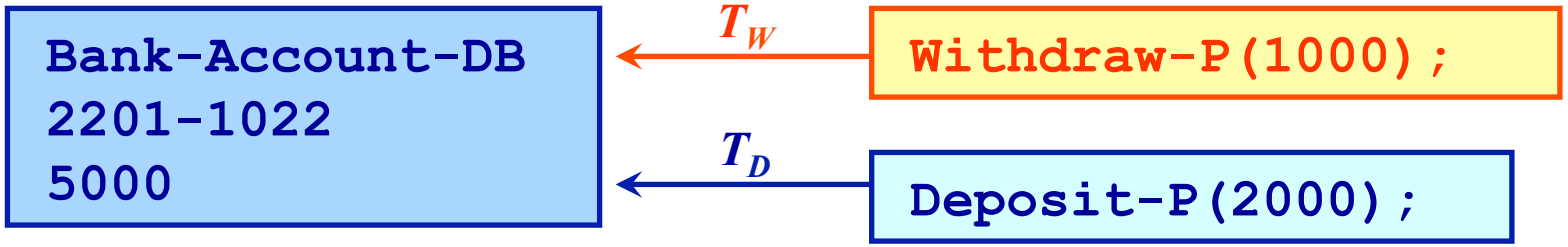
Correct balance

Concurrency Issue (1/4)

But… What if two processes accesses the *same database record at the same time*?

- **Which process get access first?**
- **Does it matter?**

Concurrency Problem (2/4)

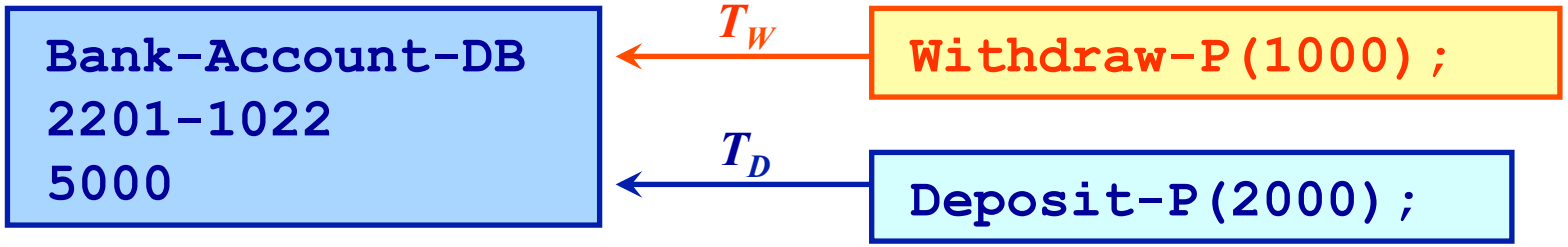


If ($T_W = T_D$), and Deposit-process "got in" first

Time	Withdraw-P(1000)	Deposit(2000)	Balance
T_D		Check balance [5000]	5000
T_D+1	Check balance [5000]		5000
T_D+2		Bal ← Bal + 2000	7000
T_D+3	Bal ← Bal - 1000		4000

Wrong balance

Concurrency Problem (2/3)



If ($T_W = T_D$), and Withdraw-process "got in" first

Time	Withdraw-P(1000)	Deposit(2000)	Balance
T_W	Check balance [5000]		5000
T_W+1		Check balance [5000]	5000
T_W+2	Bal ← Bal - 1000		4000
T_W+3		Bal ← Bal + 2000	7000

Wrong balance

Concurrency Problem (3/3)

- Operations of the two processes are *interleaved*
  - Withdraw-Process and Deposit-Process "interfere" with each other
- Wrong balance for both cases
  - Since one of the operations is over-written

Concurrency Solution: Lock operation

IDEA: If one process *P* is changing balance, make sure that other processes *do not access* the same balance until *P* is done

Solution: The process that "get-in" first, *locks up the record*.

This makes sure other processes *will not be to access* the same record.
Unlock the record *after* update is done.

Concurrency Solution: (2/4)

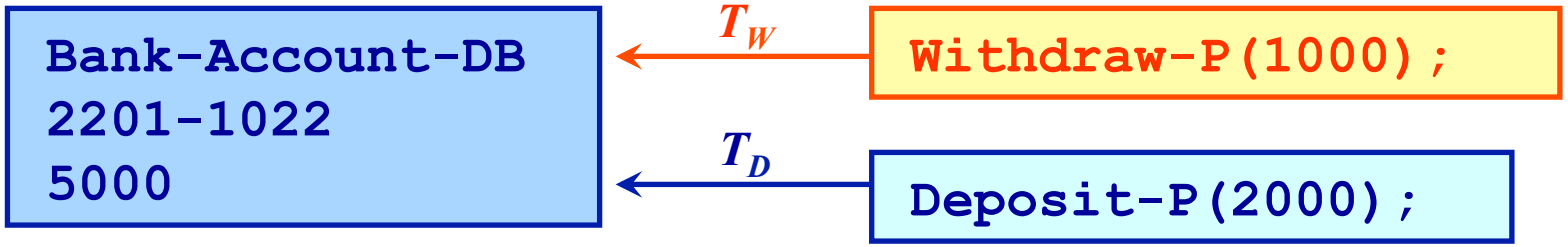
```
Bank-Account-DB
2201-1022
5000
```

❑ Deposit and Withdrawal Processes

```
Withdraw-P (amt);
begin
   Get & lock record;
   bal ← bal - amt;
   unlock record;
end;
```

```
Deposit-P (amt);
begin
   Get & lock record;
   bal ← bal + amt;
   unlock record;
end;
```

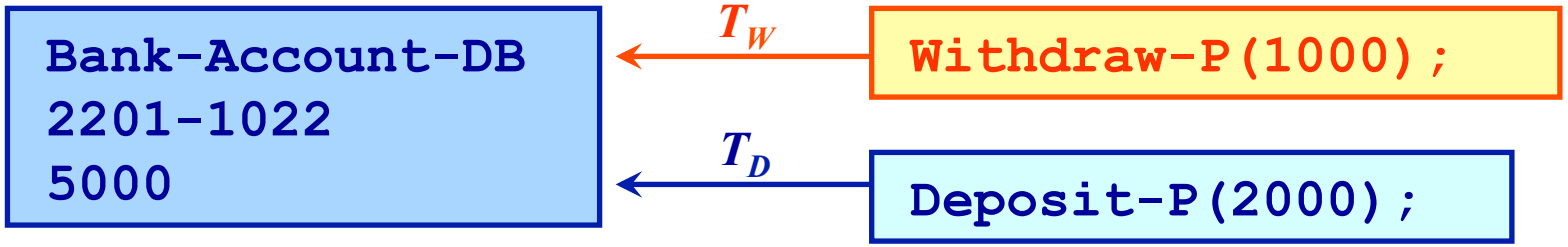
Concurrency Solution: (3/4)



If ($T_W = T_D$), and Withdraw-process "got in" first

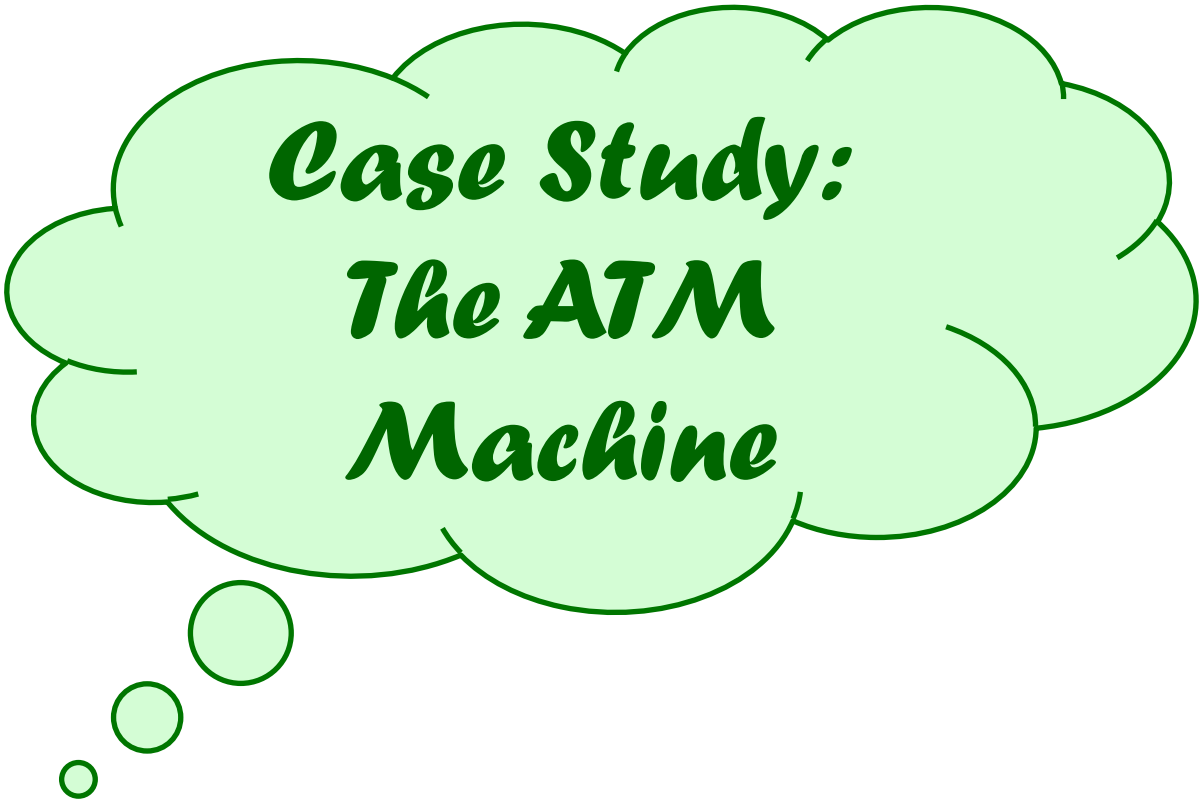
Time	Withdraw-P(1000)	Deposit(2000)	Balance
T_W	Get & Lock record;		5000
T_W+1		Get...; [blocked]	5000
T_W+2	Bal ← Bal – 1000; Unlock record;		4000
T_W+3		Get & Lock record;	4000
T_W+4		Bal ← Bal + 2000; Unlock record;	6000

Concurrency Solution: (4/4)

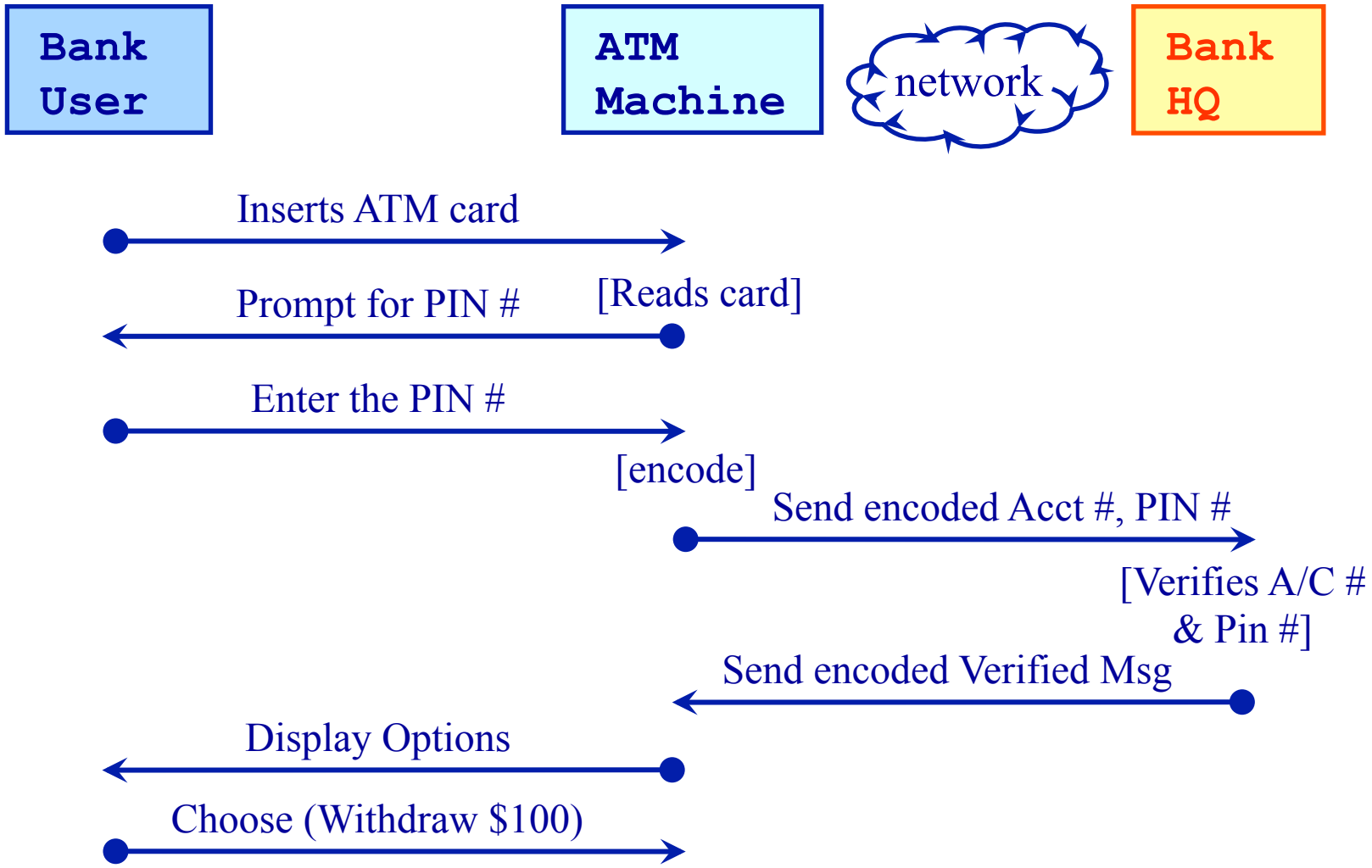


If ($T_W = T_D$), and Deposit-process "got in" first

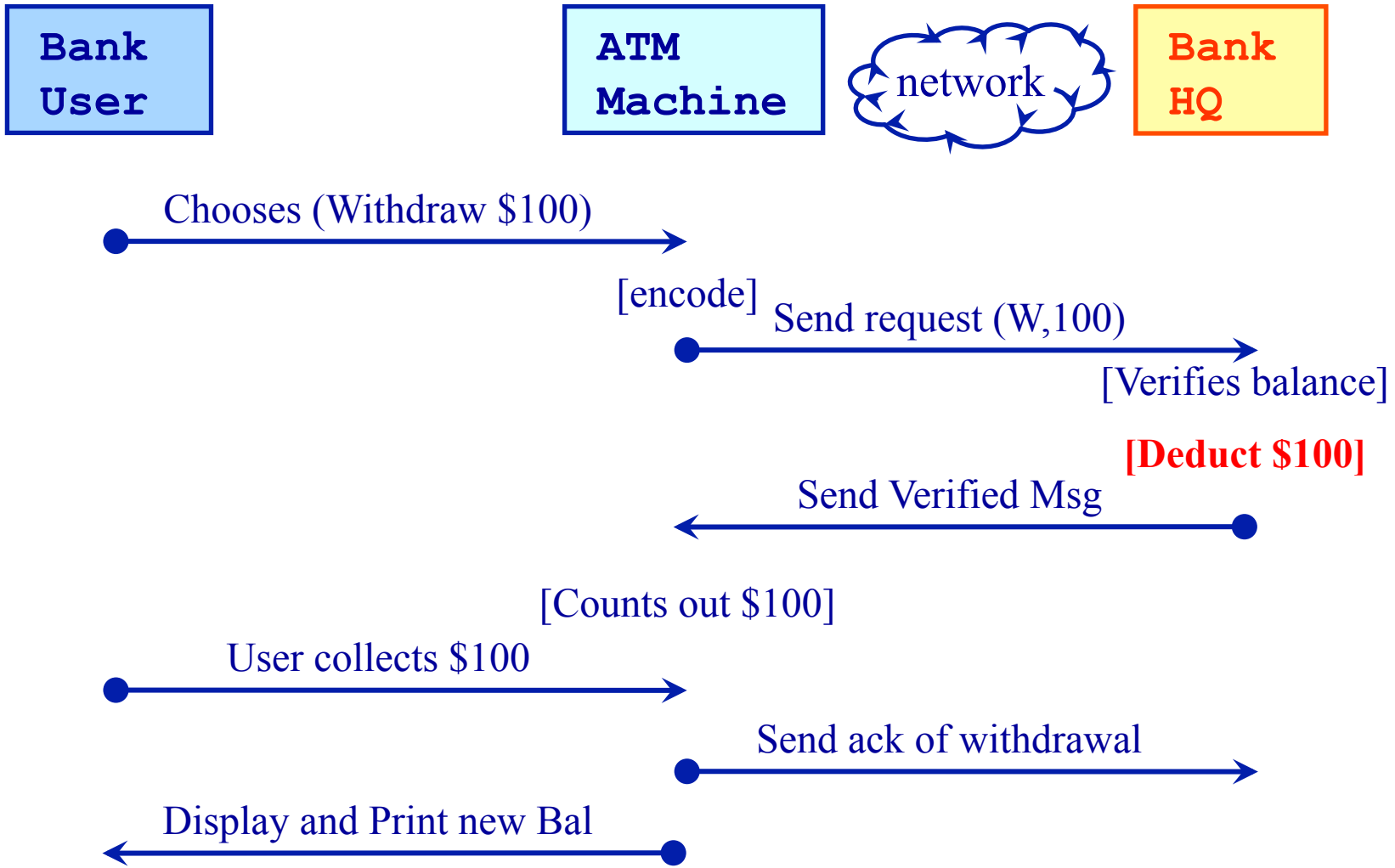
Time	Withdraw-P(1000)	Deposit(2000)	Balance
T_W		Get & Lock record;	5000
T_W+1	Get...; [blocked]		5000
T_W+2		Bal ← Bal + 2000; Unlock record;	7000
T_W+3	Get & Lock record;		7000
T_W+4	Bal ← Bal – 1000; Unlock record;		6000



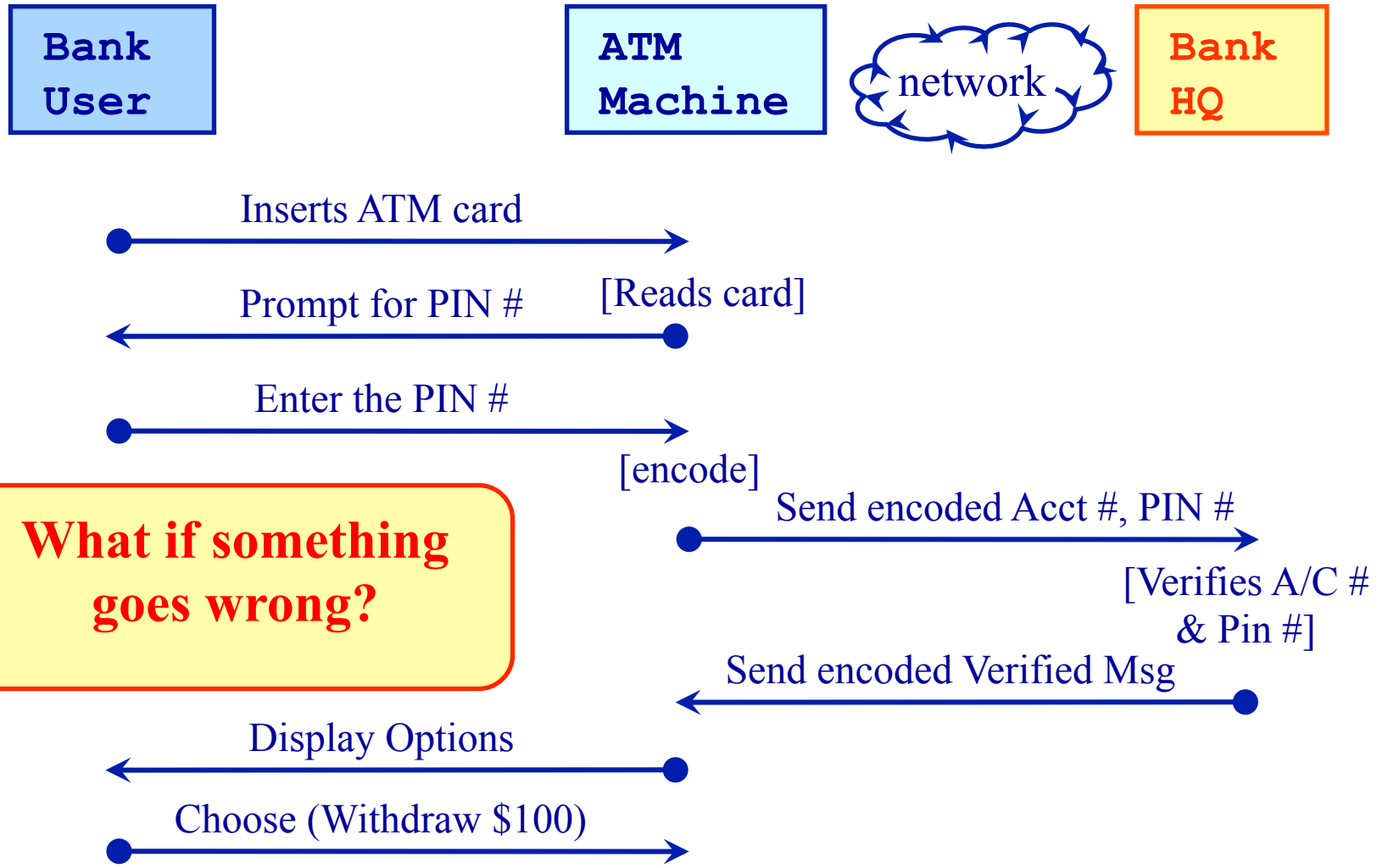
Simple ATM Scenario (1 of 2)



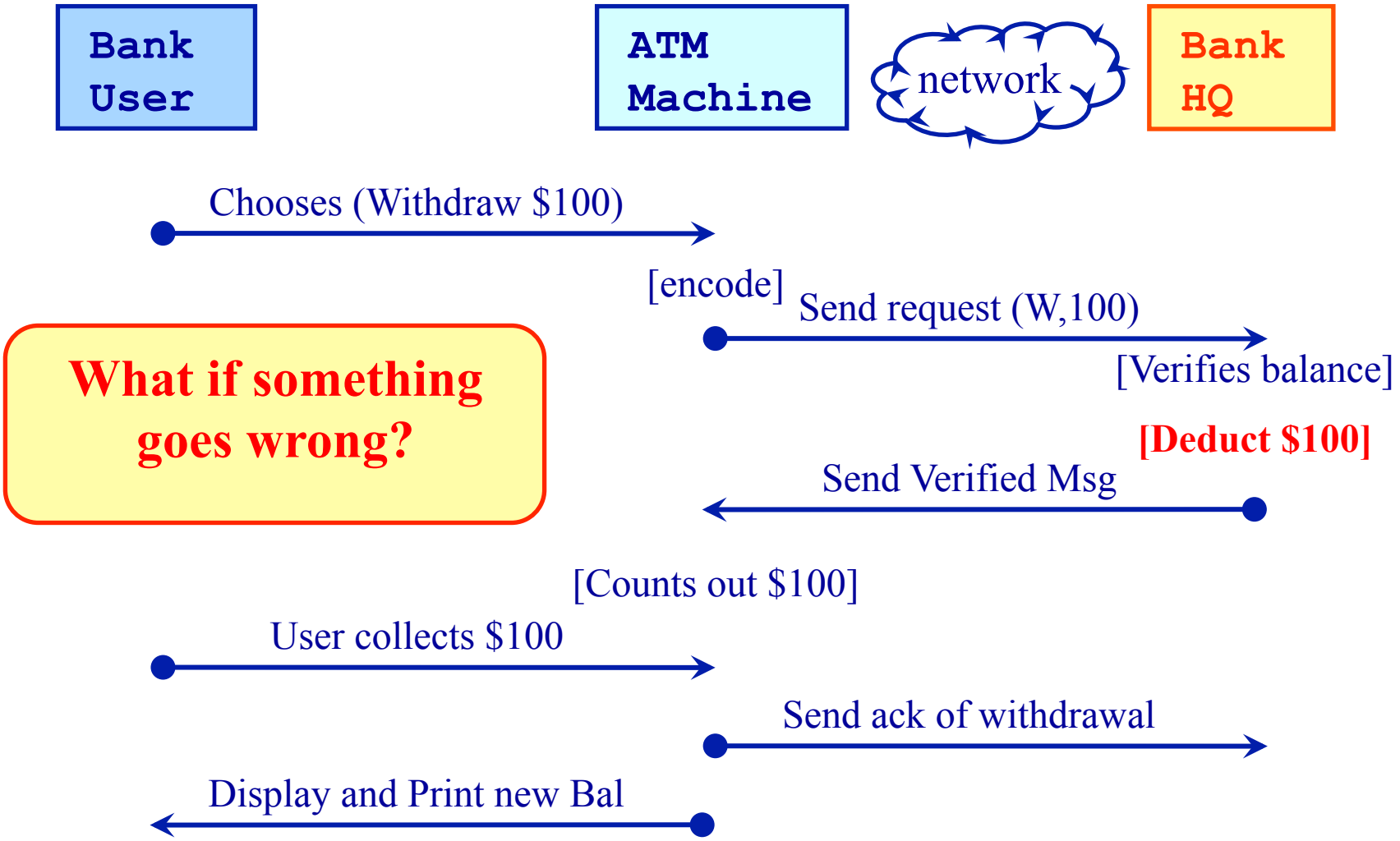
Simple ATM Scenario (2 of 2)



Simple ATM Scenario: Malfunction



Simple ATM Scenario: Malfunction…



Actually, no technical solution…

- **The ATM problem is similar to "The Two Generals Problem"**
  - https://en.wikipedia.org/wiki/Two_Generals%27_Problem

Thank you!