



Какво представлява метода *Error correction* – коригиране на грешката?

- Класически алгоритми, които адресират класически проблем – няма нищо КВАНТОВО

Какво представлява метода *Error correction* – коригиране на грешката?

- Класически алгоритми, които адресират класически проблем – няма нищо квантово
- Възникват спорадични грешки в следствие на генерирането на сигналите, разпространението им и получаването им. Тези грешки променят съдържанието на съобщението



Какво представлява метода *Error correction* – коригиране на грешката?

- Класически алгоритми, които адресират класически проблем – няма нищо квантово
- Възникват спорадични грешки в следствие на генерирането на сигналите, разпространението им и получаването им. Тези грешки променят съдържанието на съобщението



- Честота на възникване на грешките – различните честоти изискват различни решения. При квантовите системи следим параметъра *QBER* (*Quantum Bit Error Rate*)

Какво представлява метода *Error correction* – коригиране на грешката?

- Класически алгоритми, които адресират класически проблем – няма нищо КВАНТОВО
- Възникват спорадични грешки в следствие на генерирането на сигналите, разпространението им и получаването им. Тези грешки променят съдържанието на съобщението



- Честота на възникване на грешките – различните честоти изискват различни решения. При квантовите системи следим параметъра *QBER* (*Quantum Bit Error Rate*)
- Ще разгледаме няколко решения: повторение, проверка за грешки, коригиране на грешки

Коригиране на грешки чрез повторение

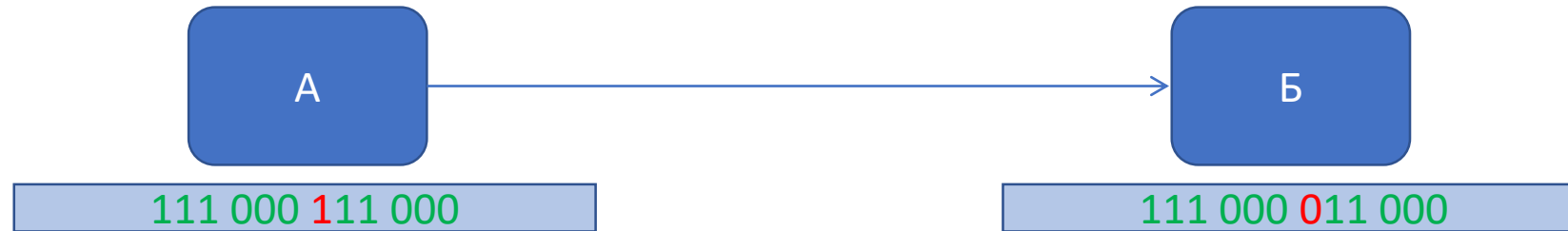
- Най-интуитивният подход – повтаряме всеки изпратен бит по 3 пъти.





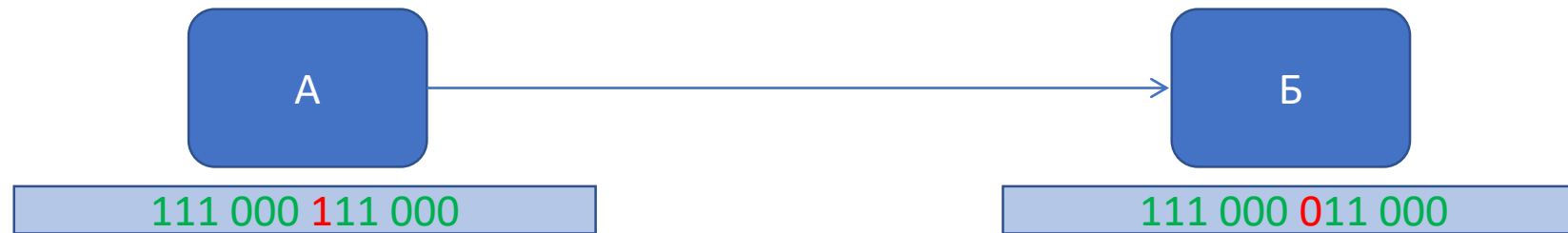
Коригиране на грешки чрез повторение

- Най-интуитивният подход – повтаряме всеки изпратен бит по 3 пъти.
- Съобщението 1010 се превръща в 111 000 111 000



Коригиране на грешки чрез повторение

- Най-интуитивният подход – повтаряме всеки изпратен бит по 3 пъти.
- Съобщението `1010` се превръща в `111 000 111 000`



- Отнема значително повече ресурси и забавя комуникацията. Може да се справи с висока честота на грешките



Проверка за грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове – не се опитваме да открием точно къде е или да я коригираме 1010



Проверка за грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове – не се опитваме да открием точно къде е или да я коригираме 1010
- Добавяме един бит към този блок, който служи за проверка. 1010 0
Този бит проверява четността на целия блок – сумираме всички битове в блока и записваме получената стойност в този допълнителен бит

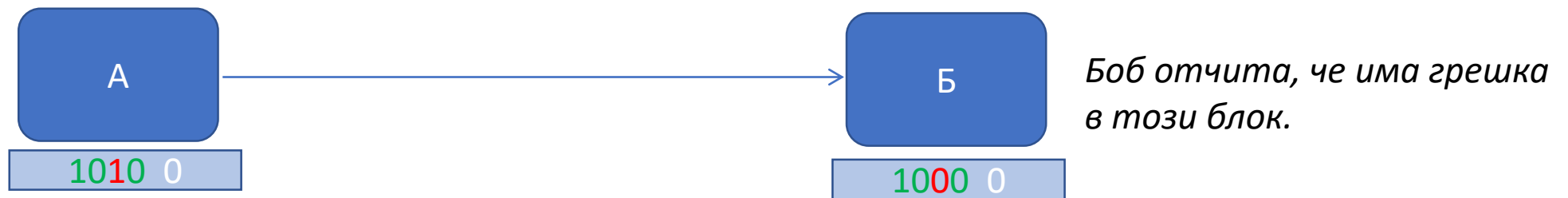


Проверка за грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове – не се опитваме да открием точно къде е или да я коригираме `1010`
- Добавяме един бит към този блок, който служи за проверка. `1010 0`
Този бит проверява четността на целия блок – сумираме всички битове в блока и записваме получената стойност в този допълнителен бит
- Изпращаме съобщението и бита за проверка. `1010 0`
При получаването Боб сумира всички битове в блока и сравнява с получения бит за проверка

Проверка за грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове – не се опитваме да открием точно къде е или да я коригираме `1010`
- Добавяме един бит към този блок, който служи за проверка. `1010 0`
Този бит проверява четността на целия блок – сумираме всички битове в блока и записваме получената стойност в този допълнителен бит
- Изпращаме съобщението и бита за проверка. `1010 0`
При получаването Боб сумира всички битове в блока и сравнява с получения бит за проверка





Коригиране на грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове, да я локализираме и да я поправим 1010



Коригиране на грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове, да я локализираме и да я поправим 1010
- Добавяме три бита към този блок, който служи за проверка. Тези битове проверява четността на части от блока по следния начин (код на Хаминг):
 - Бит за коригиране 1: бит 1 + бит 2 + бит 4
 - Бит за коригиране 2: бит 1 + бит 3 + бит 4
 - Бит за коригиране 3: бит 2 + бит 3 + бит 4 1010 101



Коригиране на грешки

- Контекстът на този проблем е, че искаме да проверим дали има грешка в един блок от битове, да я локализираме и да я поправим 1010
- Добавяме три бита към този блок, който служи за проверка. Тези битове проверява четността на части от блока по следния начин (код на Хаминг):
 - Бит за коригиране 1: бит 1 + бит 2 + бит 4
 - Бит за коригиране 2: бит 1 + бит 3 + бит 4
 - Бит за коригиране 3: бит 2 + бит 3 + бит 4 1010 101
- Изпращаме съобщението и битовете за коригиране. При получаването Боб сумира всички битове в блока и сравнява с получените битове за коригиране

Коригиране на грешки



- Боб извършва същите операции като Алис, а именно сумира битовете по следния начин:
 - Бит за коригиране 1: бит 1 + бит 2 + бит 4
 - Бит за коригиране 2: бит 1 + бит 3 + бит 4
 - Бит за коригиране 3: бит 2 + бит 3 + бит 4

110

Коригиране на грешки



- Боб извършва същите операции като Алис, а именно сумира битовете по следния начин:
 - Бит за коригиране 1: бит 1 + бит 2 + бит 4
 - Бит за коригиране 2: бит 1 + бит 3 + бит 4
 - Бит за коригиране 3: бит 2 + бит 3 + бит 4 110
- Открива разминаване между неговата проверка и получените битове за корекция от Алис – следователно има грешка в блока от битове

Коригиране на грешки



- Боб извършва същите операции като Алис, а именно сумира битовете по следния начин:
 - Бит за коригиране 1: бит 1 + бит 2 + бит 4
 - Бит за коригиране 2: бит 1 + бит 3 + бит 4
 - Бит за коригиране 3: бит 2 + бит 3 + бит 4 110
- Открива разминаване между неговата проверка и получените битове за корекция от Алис – следователно има грешка в блока от битове
- Бит за коригиране 1 има правилна стойност, следователно грешката е в бит 3 от оригиналното съобщение. Коригираното съобщение е: 1010