



# شبکه‌های خبراتی

سید حمید صفوی

دانشکده فنی و مهندسی

دانشگاه محقق اردبیلی

نیمسال دوم ۹۷-۹۸

# تصحیح خطا



# سرفصل

- برخی از بیت‌ها ممکن است به دلیل نویز به صورت خطا دریافت شوند. چگونه آن‌ها را رفع کنیم؟
- و چرا باید از تشخیص استفاده کنیم وقتی می‌توانیم از تصحیح استفاده کنیم؟





# کدهای تصحيح خطا



# چرا تصحیح خطا سخت است؟

- اگر ما بیت‌های کنترلی (Check Bits) قابل اطمینانی داشتیم، می‌توانستیم از آن‌ها برای محدود کردن موقعیت خطا استفاده کنیم.  
– سپس تصحیح آسان خواهد بود.
- اما خطا می‌تواند در **بیت‌های کنترلی** یا **بیت‌های داده** وجود داشته باشد!  
– داده‌ها (بیت‌های پیام) حتی ممکن است درست باشد.



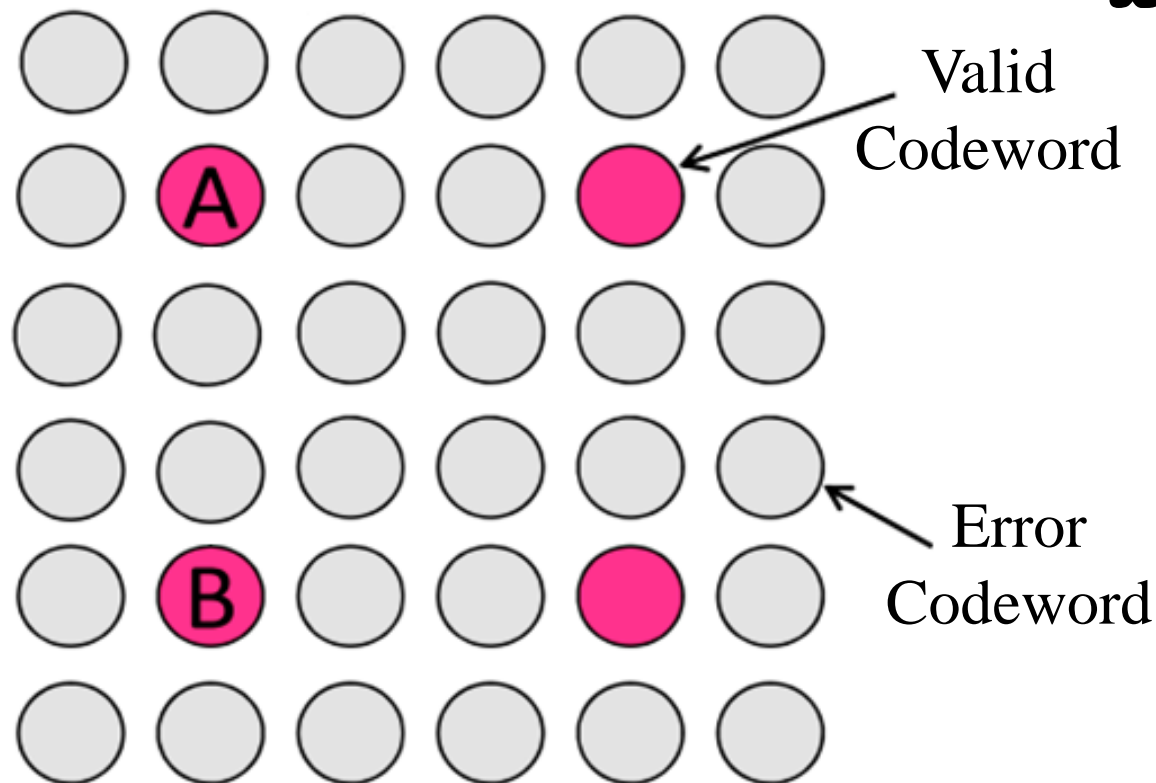
# شهود برای کدهای تصحیح خطا

- فرض کنید ما یک کد با فاصله همینگ حداقل ۳ ساخته‌ایم.
  - برای تبدیل یک کلمه کد اصلی به کلمه کد دیگر به سه بیت و یا بیش از سه بیت خطا نیاز است.
  - خطاهای تک بیتی نزدیک‌ترین مقدار به کلمه کد یکتای اصلی خواهند بود.
- اگر ما فرض کنیم که خطا **تنها** در یک بیت اتفاق افتاده است، می‌توانیم آن را به وسیله نگاشت خطا روی نزدیک‌ترین کلمه کد صحیح (معتبر) تصحیح کنیم.
  - اگر فاصله همینگ (HD) بزرگتر از  $2d+1$  باشد، برای  $d$  خطا کار می‌کند.



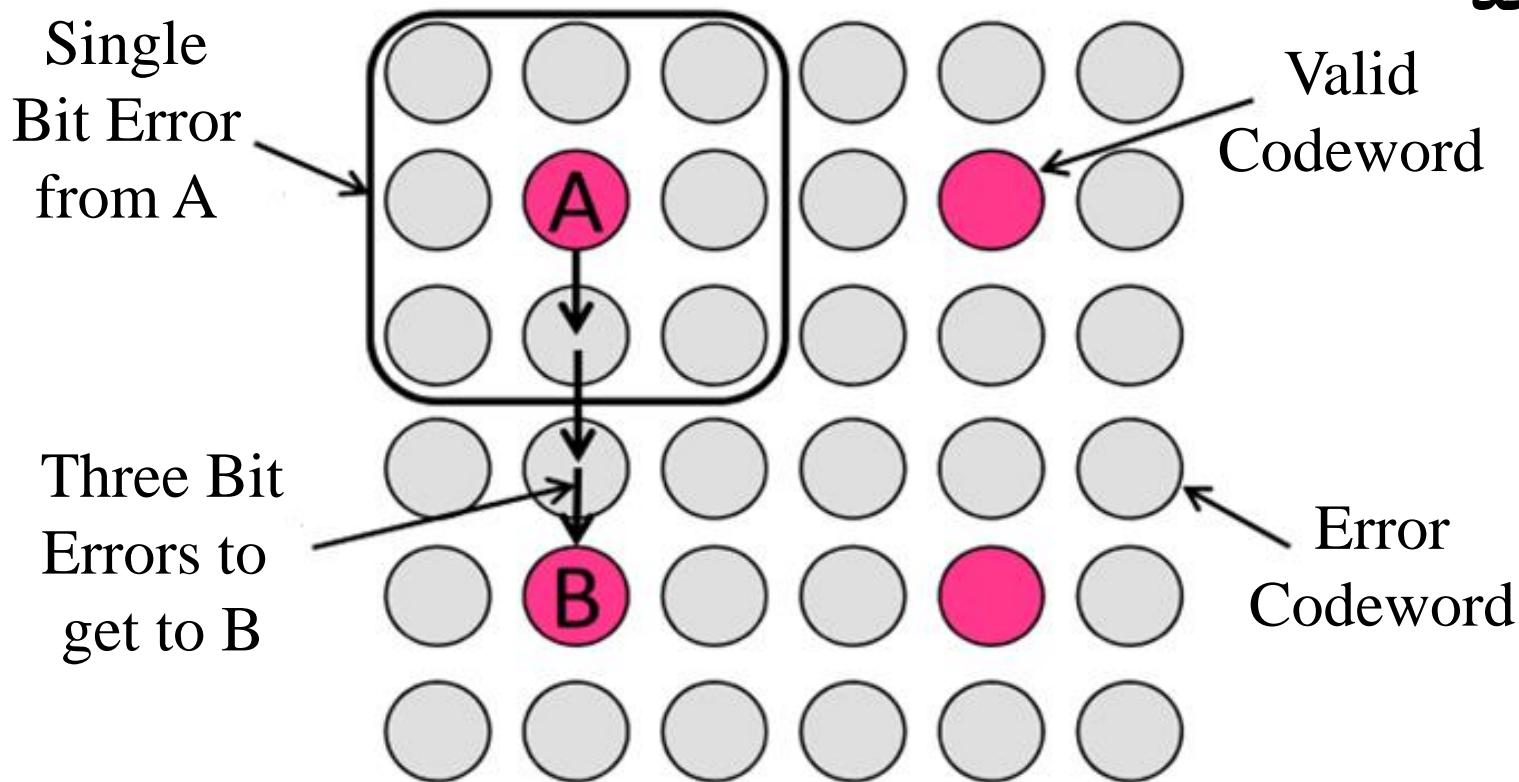
## شهود برای کدهای تصحیح خطا (۲)

• تجسم کردن کد



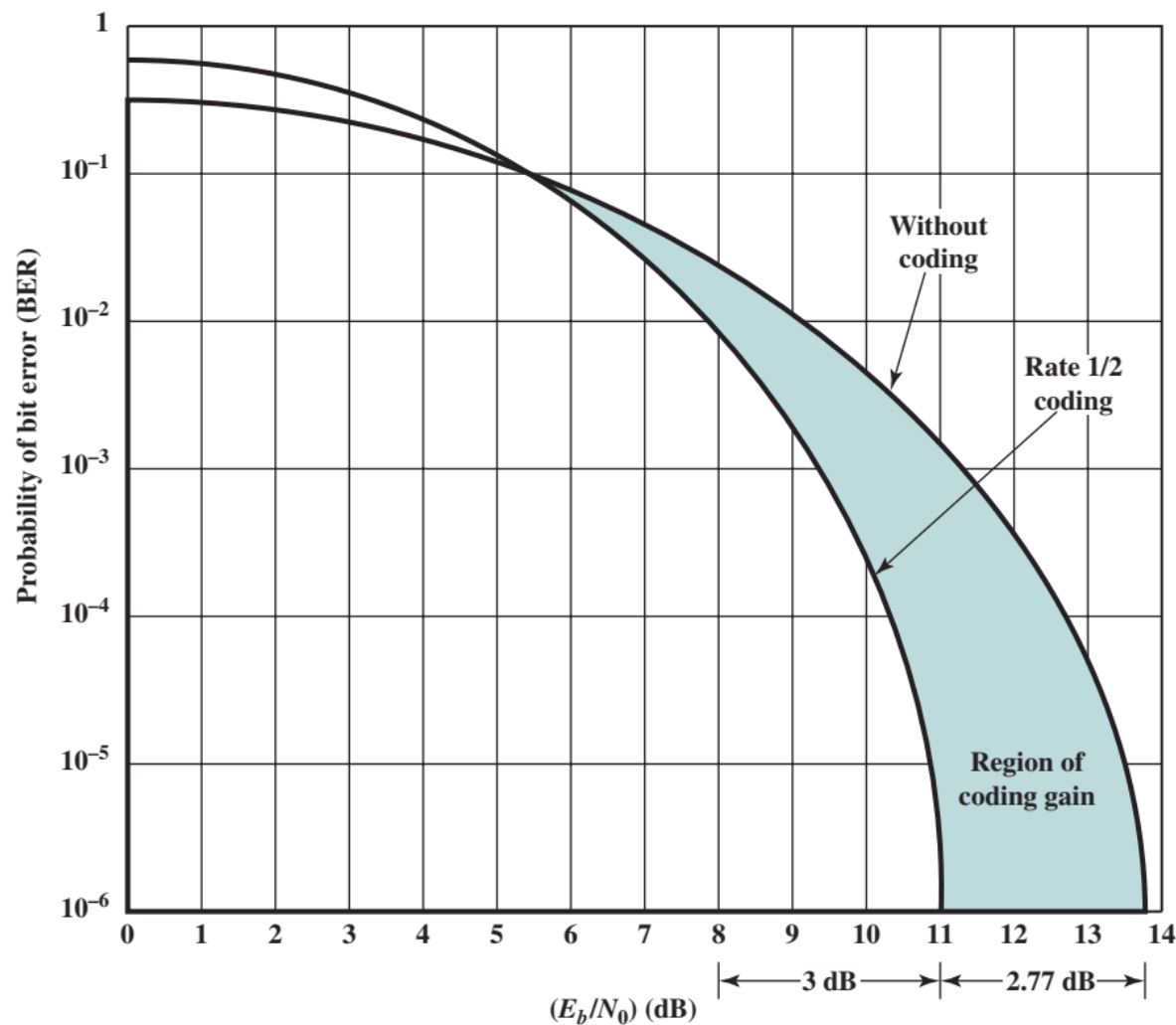
## شهود برای کدهای تصحیح خطا (۳)

- تجسم کردن کد





# کدگذاری چگونه عملکرد سیستم را بهبود می دهد؟



# کد همینگ



## کد همینگ (۲)

- در حالت کلی یک روش برای ساخت کد با فاصله همینگ  $d$  به ما می‌دهد.
  - از فرمول روبرو استفاده می‌شود؛ به عنوان مثال اگر تعداد بیت‌های کنترلی  $r = 3$  باشد، آنگاه تعداد  $m = 4$  بیت پیام را می‌توانیم ارسال کنیم.

$$m = 2^r - r - 1$$

- بیت کنترلی را در موقعیت‌های  $p$  که توان ۲ باشند، قرار می‌دهیم؛ با شروع از موقعیت ۱
- بیت کنترلی در موقعیت  $p$  برابر است با موقعیت‌هایی با معیار  $p$  که در مقادیر خود دارد.

- به علاوه یک راه آسان برای آشکارسازی! (به زودی خواهیم دید...)



## اثبات رابطه $m = 2^r - r - 1$

- فرض کنید تعداد بیت‌های پیام  $m$  باشد و تعداد بیت‌های کنترلی  $r$  باشد که به ما اجازه تصحیح ۱ بیت خطا را می‌دهد. هر کدام از  $2^m$  پیام مجاز، دارای  $n = m + r$  تعداد کلمه کد غیرمجاز با فاصله همینگ ۱ از آن است. این کلمه کدهای غیرمجاز با تغییر یک بیت از کلمه کدهای  $n$  تایی حاصل می‌شوند.
- بنابراین هر کدام از  $2^m$  پیام مجاز نیاز به  $n+1$  الگوی بیت مجزا و مختص خود دارد. از آنجایی که تعداد کل الگوی بیت برابر  $2^n$  است، بنابراین باید داشته باشیم:  $(n+1)2^m \leq 2^n$
- با استفاده از رابطه  $n = m + r$ ، رابطه فوق به صورت زیر ساده می‌شود:
$$m \leq 2^r - r - 1$$
- با داشتن  $m$ ، حد پایین تعداد بیت کنترلی لازم به دست می‌آید.



## کد همینگ (۳)

• به عنوان مثال:

اگر  $\text{data} = 0101$  و  $k = 3$  باشد، یعنی سه بیت کنترلی داشته باشیم، آنگاه:

- در یک کد هفت بیتی، بیت‌های کنترلی دارای موقعیت‌های ۱، ۲، ۴ خواهند بود.
- بیت کنترلی ۱ به وسیله موقعیت‌های ۱، ۳، ۵، ۷ به وجود می‌آید. (001-011-101-111)
- بیت کنترلی ۲ به وسیله موقعیت‌های ۲، ۳، ۶، ۷ به وجود می‌آید. (010-011-110-111)
- بیت کنترلی ۴ به وسیله موقعیت‌های ۴، ۵، ۶، ۷ به وجود می‌آید. (100-101-110-111)

0 1 0 0 1 0 1      →  
1 2 3 4 5 6 7

$$p_1 = 0 + 1 + 1 = 0, \quad p_2 = 0 + 0 + 1 = 1, \quad p_4 = 1 + 0 + 1 = 0$$





## کد همینگ (۴)

- برای کدبرداری:
  - محاسبه مجدد بیت‌های کنترلی (به وسیله جمعی که شامل خود چک‌بیت‌ها هم می‌شود)
  - مرتب‌کردن به عنوان یک عدد باینری
  - مقدار سندروم را به دست می‌آوریم، که موقعیت خطای رخ داده را به ما می‌دهد.
  - مقدار سندروم صفر به این معنی است که خطایی رخ نداده‌است.
  - در غیر این صورت، با تغییر بیت (flip)، خطا اصلاح می‌شود.



## کد همینگ (۵)

• ادامه مثال قبل:

<u>0</u>	<u>1</u>	0	<u>0</u>	1	0	1
1	2	3	4	5	6	7

$$p_1 = 0 + 0 + 1 + 1 = 0, \quad p_2 = 1 + 0 + 0 + 1 = 0, \quad p_4 = 0 + 1 + 0 + 1 = 0$$

Syndrome = 000, no error

Data = 0101



## کد همینگ (۶)

• ادامه مثال قبل:

<u>0</u>	<u>1</u>	0	<u>0</u>	1	1	1
1	2	3	4	5	6	7

$$p_1 = 0 + 0 + 1 + 1 = 0, \quad p_2 = 1 + 0 + 1 + 1 = 1, \quad p_4 = 0 + 1 + 1 + 1 = 1$$

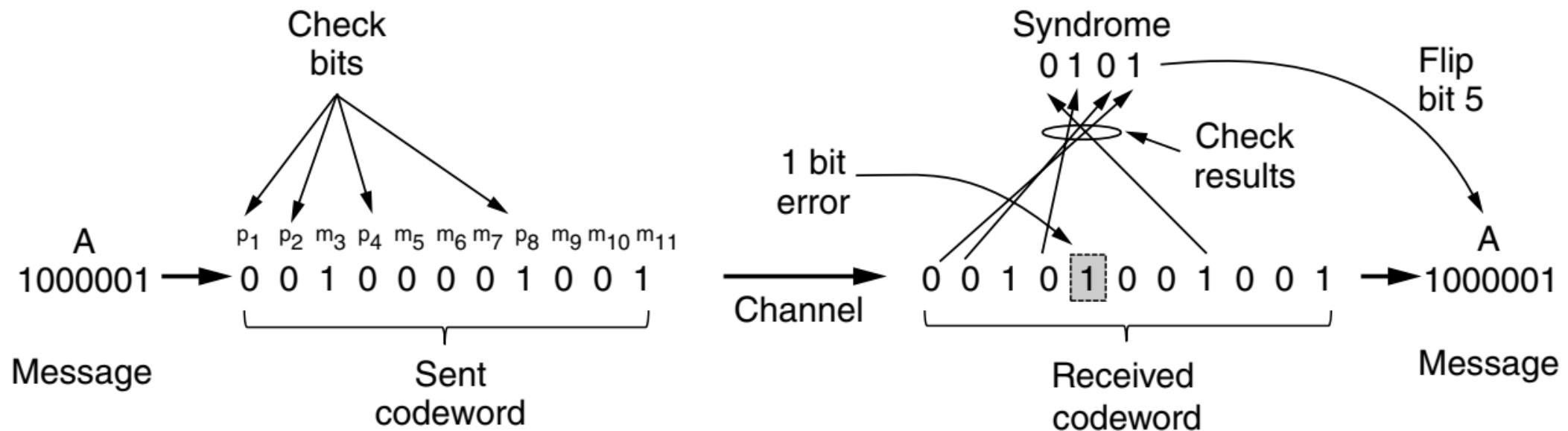
Syndrome = **110**, flip position 6

Data = **0101** (correct after flip!)



## کد همینگ (۷)

• مثال دیگر:



# دیگر کدهای تصحیح خطا - کدهای کانولوشنی

- کدهایی که در عمل استفاده می‌شوند بیشتر بر اساس همینگ طراحی می‌شوند.

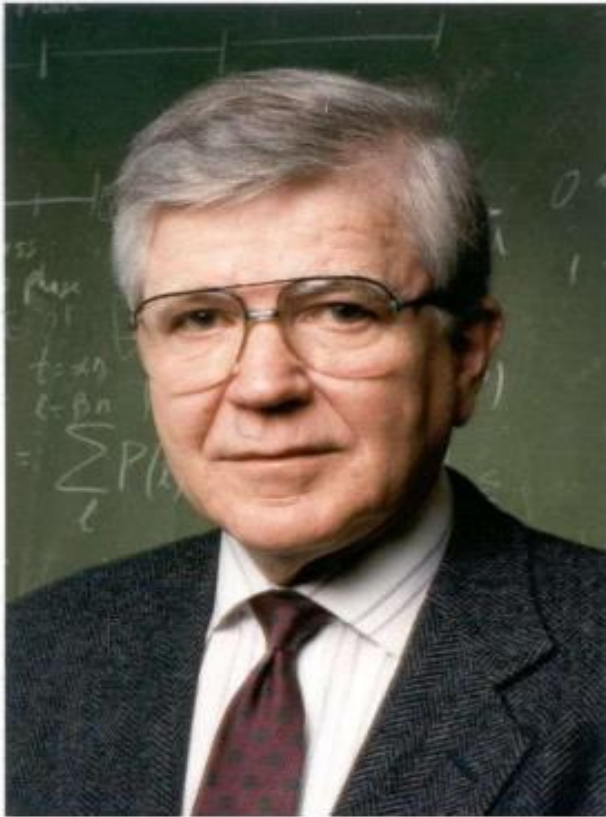
- کدهای کانولوشنی:

- جریانی از داده‌ها را می‌گیرد و ترکیبی از بیت‌های دریافتی اخیر را در خروجی می‌دهد.
- باعث می‌شود شکنندگی خروجی کمتر شود.
- برای کدبرداری از الگوریتم ویتربی (Viterbi) استفاده می‌شود. (که می‌تواند از مقدار بیت اطمینان استفاده کند)





# دیگر کدهای تصحیح خطا – LDPC



Source: IEEE GHN, © 2009 IEEE



- Low density parity check
  - LDPC بر اساس ماتریس های تُنک
  - کدبرداری با روش تکراری و با استفاده از الگوریتم Belief Propagation Algorithm
  - یک روش به روز و کارآمد
- اختراع شده توسط Robert Gallager در سال ۱۹۶۳ به عنوان بخشی از پایان نامه دکتری خود
  - به سرعت تا سال ۱۹۹۶ فراموش شده بود...

# تشخیص در برابر تصحیح

- این که کدام یک بهتر است به **الگوی خطا** بستگی دارد. به عنوان مثال:
  - ۱۰۰۰ بیت پیام با نرخ خطای بیت (BER) ۱ در ۱۰۰۰۰
- کدام overhead کمتری دارد؟
  - این هم هنوز بستگی به الگوی خطا دارد و نیاز است که اطلاعات بیشتری درباره خطاها داشته باشیم ...



## تشخیص در برابر تصحیح (۲)

۱. فرض کنید خطاهای بیت به صورت تصادفی هستند.

– پیام ها دارای ۰ و یا ۱ خطا هستند.

• تصحیح خطا:

– به تقریباً ۱۰ بیت کنترلی در هر پیام نیاز است.  $n = 2^k - k - 1$

– Overhead: ۱۰

• تشخیص خطا:

– به تقریباً ۱ بیت کنترلی در هر پیام به علاوه ۱۰۰۰ بیتی که برای مثال در یک دهم از زمان دوباره منتقل شده است، نیاز است.

– Overhead:  $1 + \frac{1000}{10} = 101$



## تشخیص در برابر تصحیح (۳)

۲. فرض کنید خطاها در انفجار ۱۰۰ تایی به وجود آمده است:

– تنها ۱ یا ۲ پیام در بین ۱۰۰۰ پیام دارای خطا هستند.

• تصحیح خطا:

– خیلی بیشتر از ۱۰۰ بیت کنترلی در هر پیام نیاز است.

– Overhead بزرگتر از ۱۰۰؟

• تشخیص خطا:

– به ۳۲ بیت کنترلی در هر پیام به علاوه ۱۰۰۰ بیتی که در دو هزارم زمان دوباره فرستاده شده است نیاز است.

$$32 + \frac{1000}{1000} \times 2 = 34 \quad \text{Overhead:}$$



## تشخیص در برابر تصحیح (۴)

### • تصحیح خطا:

- زمانی که احتمال وقوع خطا **زیاد** است، مورد نیاز است.
- یا هنگامی که زمانی برای ارسال مجدد وجود ندارد، مورد نیاز است.

### • تشخیص خطا:

- زمانی که احتمال وقوع خطا **کم** است، کارآمدتر است.
- و زمانی که خطاها در هنگام وقوع، **بزرگ** هستند.





# تصحیح خطا در عمل

- اکثراً در لایه فیزیکی استفاده می‌شوند.
  - LDPC در آینده بیشتر مورد استفاده قرار خواهد گرفت، برای لینک‌های درخواست مانند 802.11, DVB, WiMAX, LTE, و خطوط قدرت و ... استفاده می‌شوند.
  - کدهای کانولوشنی به طور گسترده‌ای در عمل استفاده می‌شود.
- تشخیص خطا (به همراه ارسال مجدد) در لایه لینک و لایه‌های بالاتر برای خطاهای باقی مانده استفاده می‌شود.
- تصحیح خطا همچنین در لایه کاربرد استفاده می‌شود.
  - مانند تصحیح خطای رو به جلو (Forward Error Correction (FEC
  - معمولاً با مدل خطای erasure
  - برای مثال Reed-Solomon (CDs, DVDs, etc.)

