**Mining of Massive Dataset**

# **Locality – Sensitive Hashing**

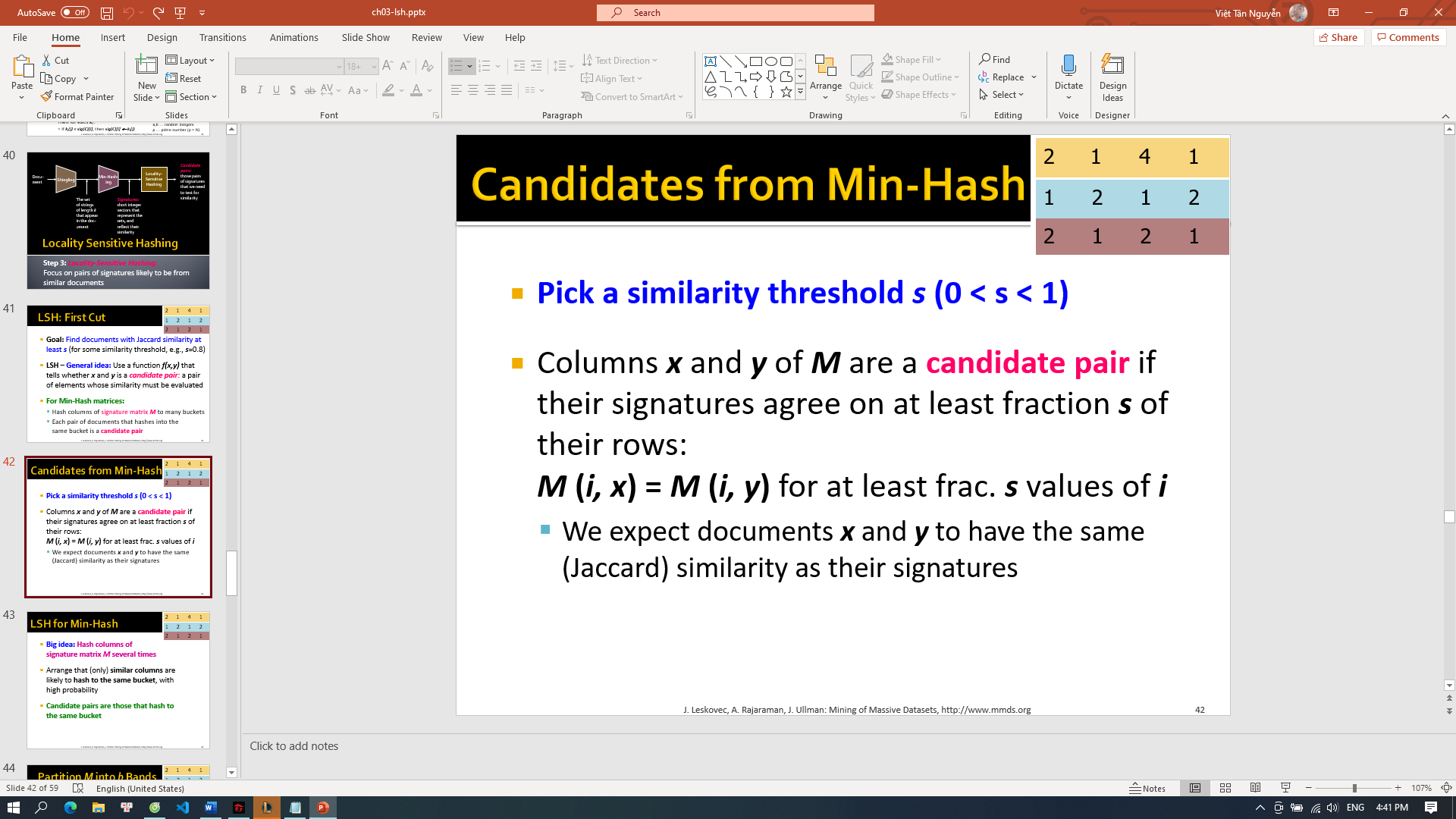
## **Candidates from Min-Hash**

Người ta đã chuyển đổi các tài liệu thành các bộ bệnh zona, và sau đó các bộ bệnh zona có lẽ lớn được chuyển đổi thành các chữ ký ngắn, thường là các vectơ của số nguyên. Chúng ta có thể so sánh hai chữ ký vì các bộ cơ bản của chúng có độ tương đồng Jaccard cao. Vì các chữ ký ngắn, chúng ta có thể đưa một số lượng lớn chúng vào bộ nhớ chính cùng một lúc, cho phép chúng ta so sánh một số cặp chữ ký khác nhau mà không cần phải đọc từng chữ ký từ đĩa nhiều lần.

Khái niệm đằng sau LSH là xem xét một tập hợp các phần tử, trong trường hợp này là các chữ ký, có các cặp liên quan mà chúng ta muốn tìm và tạo ra một danh sách ngắn các cặp ứng cử viên có độ tương đồng phải được đo lường mà không cần phải xây dựng tất cả các cặp của các phần tử đó. Khi xây dựng các cặp ứng cử viên, bạn chỉ xem xét các yếu tố riêng lẻ, không nhìn vào bản thân các cặp.

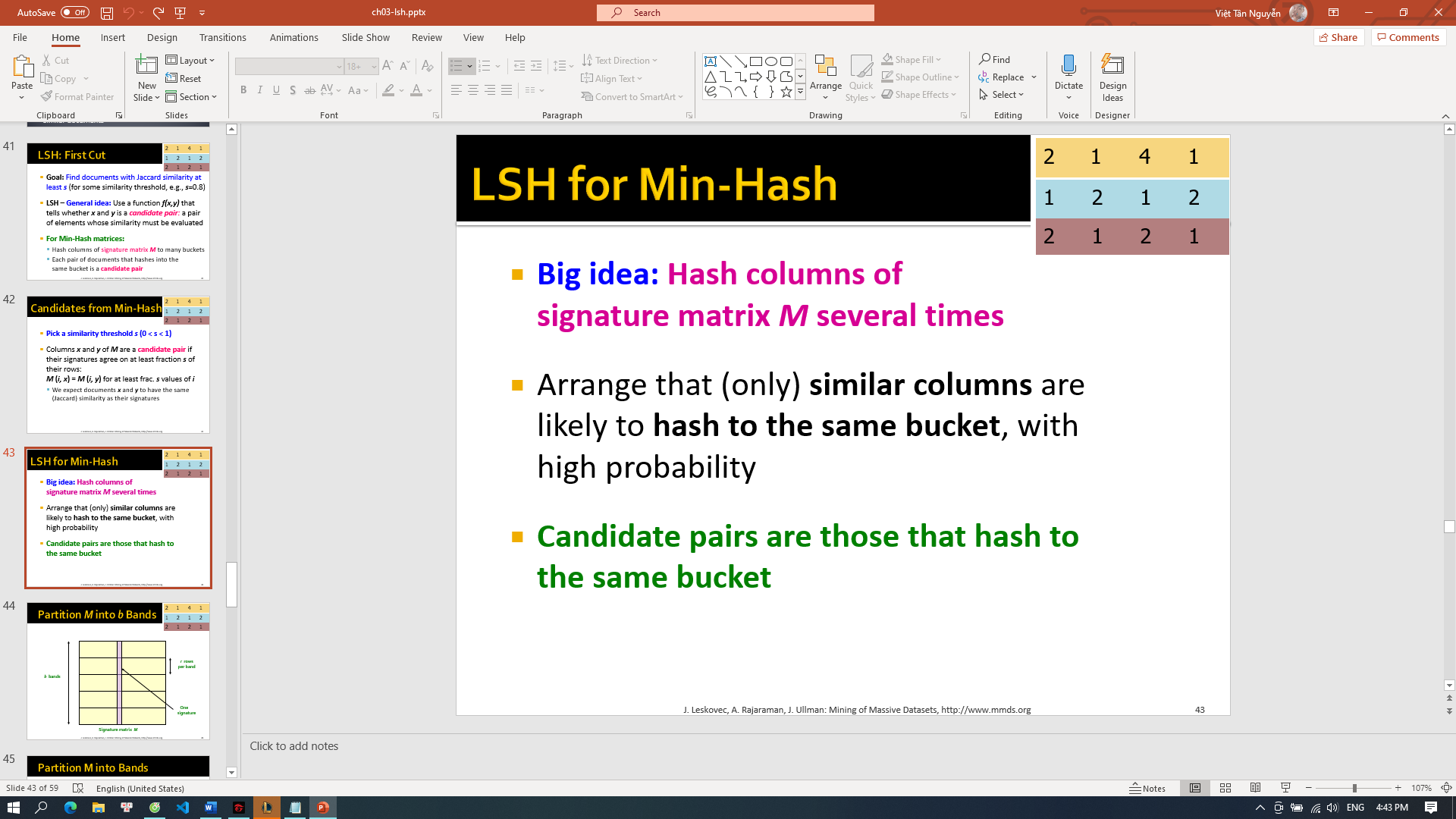
Tất cả các cặp không phải là ứng cử viên được coi là không giống nhau mặc dù trong một số trường hợp hiếm hoi, sẽ thực sự có âm tính giả. Đó là, các cặp tương tự nhưng không bao giờ được kiểm tra về độ giống nhau.

Đối với trường hợp ma trận chữ ký, chúng ta thực hiện LSH bằng cách tạo một số lượng lớn các hàm băm. Đây là các hàm băm thông thường, không phải hàm minhash. Đối với mỗi hàm băm đã chọn, chúng ta băm các cột thành nhóm. Đối với mỗi nhóm, chúng ta đặt tất cả các cặp trong nhóm đó thành một cặp ứng cử viên. Một cặp trở thành một cặp ứng cử viên nếu bất kỳ một hoặc nhiều hàm băm nào đặt cả hai chữ ký vào cùng một nhóm. Chúng ta cần điều chỉnh số lượng hàm băm và số lượng nhóm cho mỗi hàm băm để các nhóm có tương đối ít chữ ký trong đó. Bằng cách đó, không có quá nhiều cặp ứng cử viên được tạo ra. Nhưng chúng ta không thể sử dụng quá nhiều nhóm, nếu không, các cặp thực sự giống nhau sẽ không kết thúc trong cùng một nhóm cho ngay cả một trong các hàm băm mà chúng ta sử dụng.



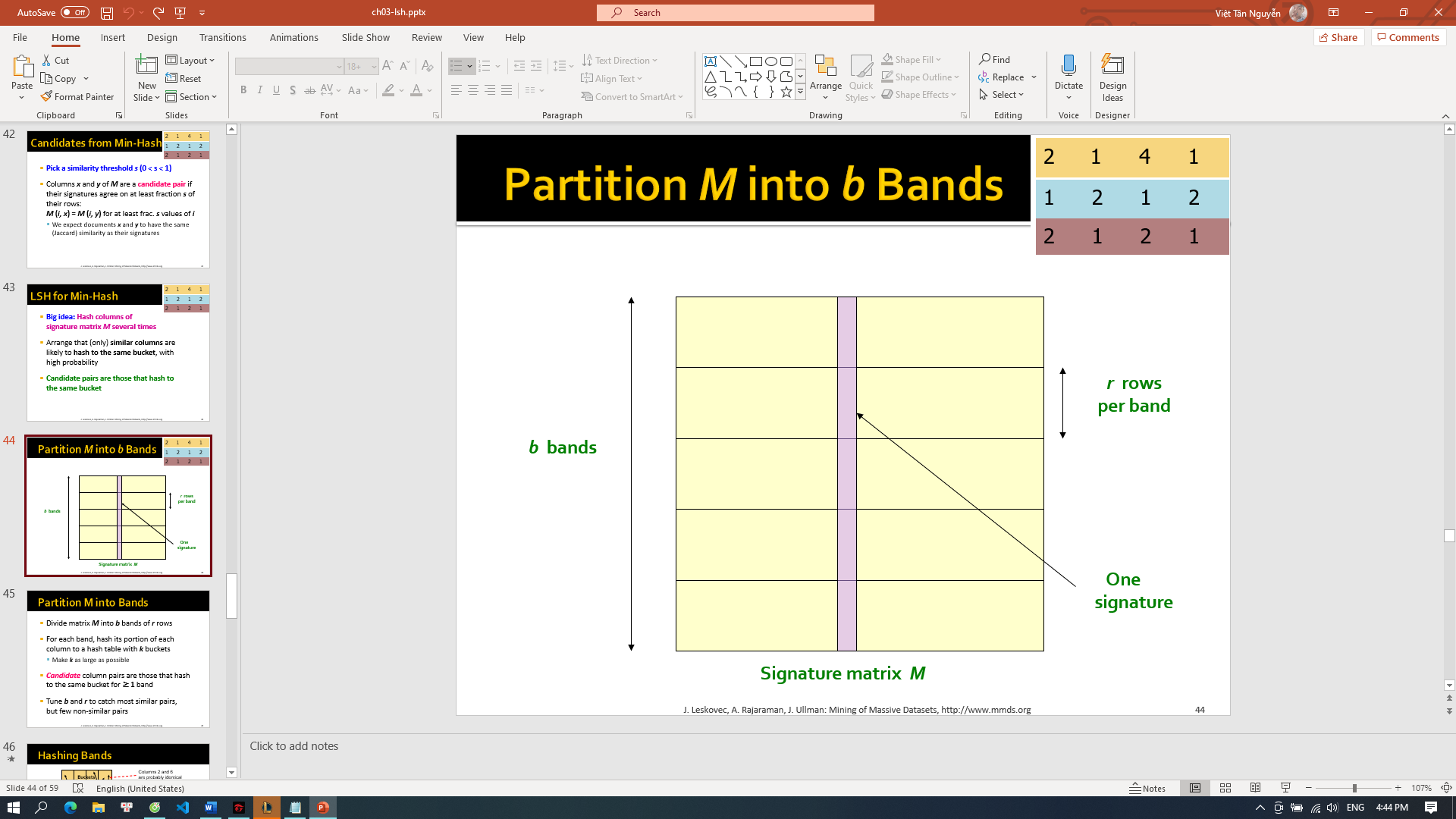
Để bắt đầu, chúng ta phải thống nhất với nhau về sự tương tự như thế nào. Chúng ta chọn ngưỡng s là giá trị tối thiểu của độ tương tự Jaccard để chúng ta coi một cặp chữ ký là tương tự. Nghĩa là, trong thế giới lý tưởng, cột x và y của ma trận chữ ký M sẽ là một cặp ứng viên nếu và chỉ khi độ tương tự của chúng ít nhất là s.

Hãy nhớ rằng sự giống nhau của các chữ ký là phần nhỏ của các thành phần hoặc các hàng của ma trận chữ ký M mà chúng đồng ý với nhau. Vì vậy, chúng ta muốn cột x và y là một cặp ứng viên nếu phần của hàng i mà m của i và x và m của i và y là ít nhất s.



Chúng ta cần tạo một số hàm băm và sử dụng mỗi hàm để băm các cột của ma trận chữ ký M thành các nhóm. Và chúng ta cần một mẹo để đảm bảo rằng các chữ ký hoặc cột tương tự, có nhiều khả năng được băm vào cùng một nhóm cho một trong các hàm băm này hơn là nếu các chữ ký không giống nhau.

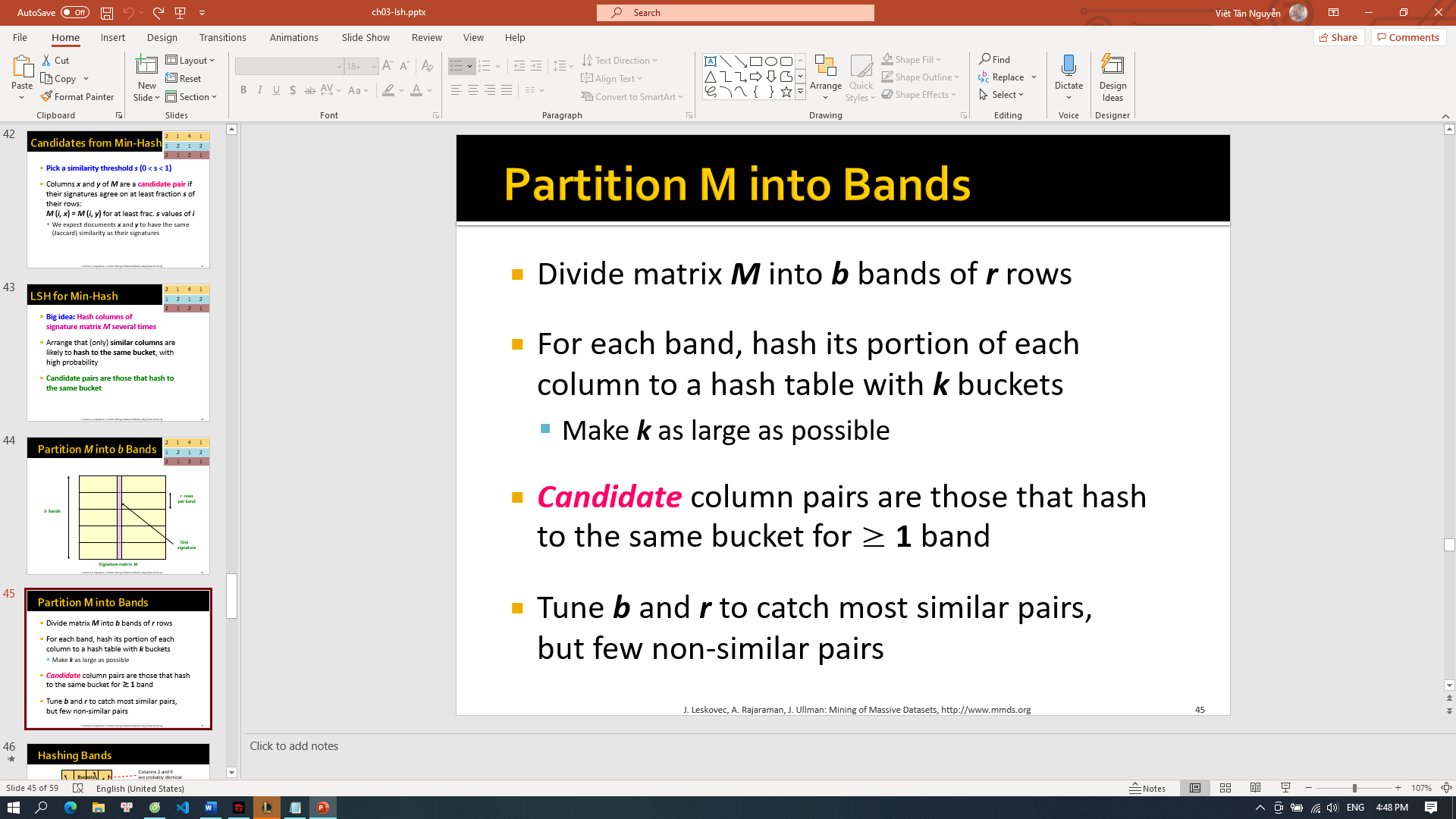
Như chúng ta đã đề cập trước đây, chúng ta sẽ coi một cặp chữ ký là một cặp ứng cử viên nếu ngay cả một trong các hàm băm đặt chúng vào cùng một nhóm.



Vì vậy, đây là cách các hàm băm được tạo ra. Vùng màu vàng là ma trận chữ ký M. Mỗi cột tương ứng với một chữ ký và mỗi hàng là một trong những thành phần của tất cả các chữ ký.

Đó là, mỗi hàng được tạo bằng cách áp dụng cho mỗi bộ cơ bản một trong các hàm minhash mà chúng ta sử dụng để tạo chữ ký ngay từ đầu. Chúng ta chia các hàng thành các dải b cho một số b.

Kết quả là có r hàng trên mỗi dải, trong đó b nhân với r là tổng chiều dài của các chữ ký. Đó là, số lượng các hàm băm chính mà chúng ta sử dụng để tạo chữ ký. Chúng ta sẽ tạo một hàm băm từ mỗi băng tần.

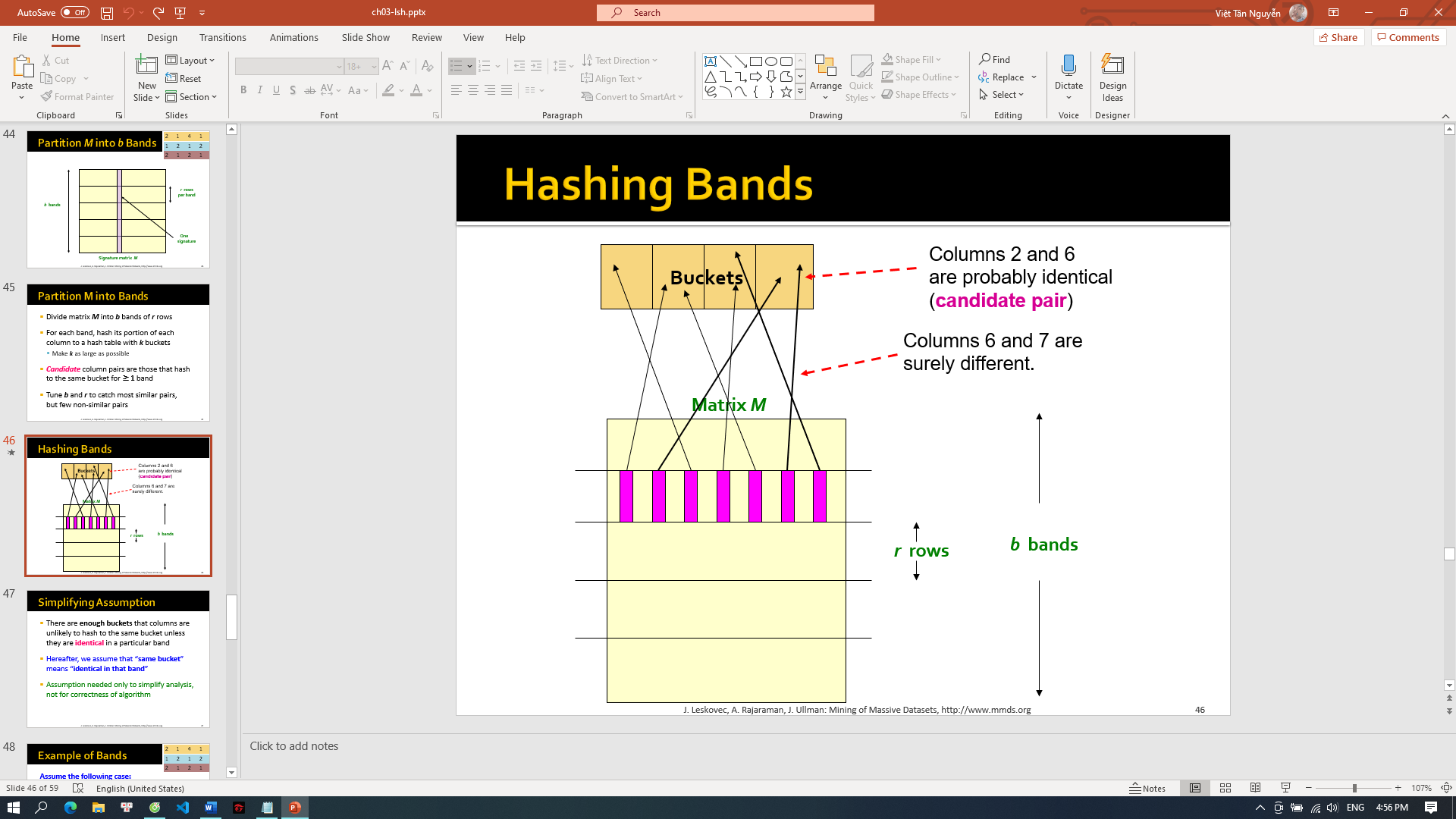


Hãy nhớ rằng, chúng ta đã chia ma trận chữ ký M thành b dải r hàng mỗi dải. Từ mỗi dải, chúng ta tạo một hàm băm. Hàm băm này chỉ băm các giá trị mà một cột nhất định có trong dải đó. Lý tưởng nhất, chúng ta sẽ tạo một nhóm cho mỗi vectơ có thể có giá trị b mà một cột có thể có trong dải đó. Đó là, chúng ta muốn có quá nhiều nhóm để hàm băm thực sự là hàm nhận dạng, nhưng đó có lẽ là quá nhiều nhóm.

Ví dụ: Nếu b bằng 5 và các thành phần của chữ ký là số nguyên 32 bit, thì chúng sẽ bằng 2 đến 5 lần 32 hoặc 2 đến lũy thừa thứ 160 của nhóm. Chúng ta thậm chí không thể nhìn vào tất cả các nhóm này để xem rốt cuộc có gì trong chúng. Vì vậy, chúng ta có thể muốn chọn một số nhóm nhỏ hơn, chẳng hạn như một triệu hoặc một tỷ. Như chúng ta đã nói, chúng ta coi một cặp cột và chữ ký là một cặp ứng cử viên nếu chúng nằm trong cùng một nhóm theo hàm băm cho bất kỳ dải nào.

Nói một cách khác, cách duy nhất chúng ta có thể chắc chắn rằng một cặp chữ ký sẽ trở thành một cặp ứng cử viên là nếu chúng, nếu chúng có chính xác các thành phần giống nhau trong ít nhất một trong các dải. Lưu ý rằng nếu hầu hết các thành phần của hai chữ ký đồng ý, thì rất có thể họ sẽ đồng ý 100% trong một ban nhạc nào đó. Nhưng nếu chúng có ít thành phần chung, thì họ không chắc sẽ đồng ý 100% trong bất kỳ ban nhạc nào. Với s, ngưỡng tương tự Jaccard cần thiết cho các cặp được coi là tương tự, chúng ta cần điều chỉnh b và r sao cho rằng hầu hết các cặp tương tự giống nhau 100% trong ít nhất một dải. Nhưng một số cặp có độ giống nhau về Jaccard ít hơn s giống nhau 100% trong bất kỳ dải nào. Hạn chế duy nhất mà chúng ta có là b lần r phải bằng độ dài của các chữ ký. Nghĩa là, bằng với số hàm minhash mà chúng ta được sử dụng để tạo chữ ký ngay từ đầu.

Theo trực giác, nếu chúng ta tạo b lớn và r nhỏ, thì sẽ có rất nhiều dải và do đó rất nhiều cơ hội cho một cặp giao lưu trong cùng một nhóm. Và vì r, chiều rộng của dải, nhỏ nên không khó để một cặp để băm vào cùng một nhóm cho một trong các dải. Vì vậy, làm cho b lớn là tốt trong tương tự, nếu ngưỡng tương tự là tương đối thấp. ngược lại, nếu bạn làm cho b nhỏ và r lớn, thì sẽ rất khó để hai chữ ký để băm vào cùng một nhóm cho một ban nhạc nhất định và có một vài ban nhạc cho họ cơ hội để làm điều đó. Vì vậy, một số lượng nhỏ các dải là tốt nhất nếu chúng ta có ngưỡng tương đồng cao. Một lần nữa, chúng ta sẽ sớm làm cho phép toán trở nên chính xác.

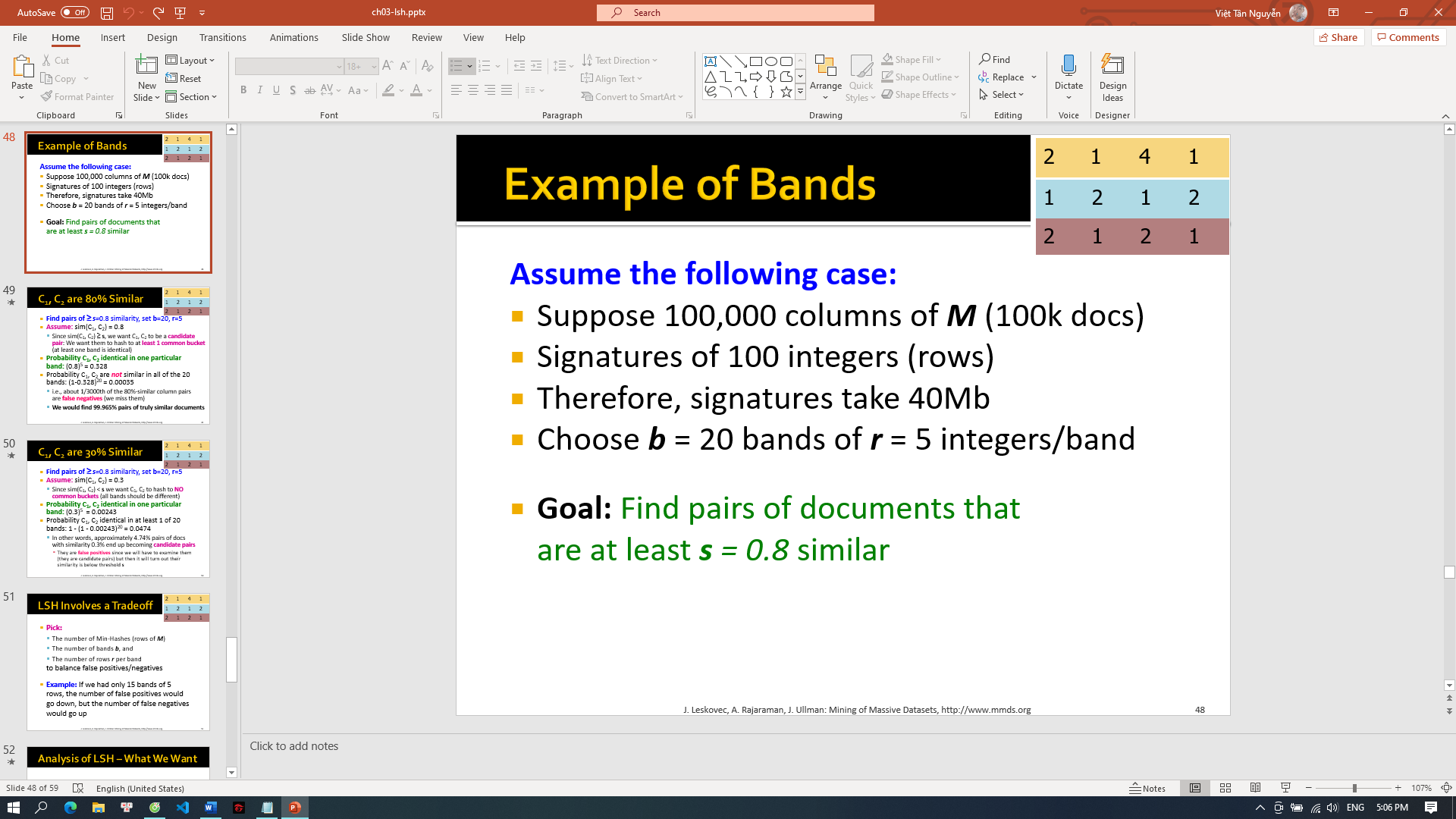


Trước khi chúng ta tiếp tục, đây là hình ảnh về những gì một trong những hàm băm cho LSH trên ma trận chữ ký trông như thế nào. Chúng ta thấy một trong những dải b, tất nhiên là dải bao gồm r hàng. Nó cũng hiển thị ma trận bao gồm một số, trong số bảy cột hoặc chữ ký.

Và mỗi đại diện màu tím, hình chữ nhật đại diện cho phần của cột của nó trong một dải mà chúng ta tập trung vào. Bây giờ, cột sáu và bảy băm thành các nhóm khác nhau. Do đó, chúng chắc chắn khác nhau trong dải này, vì vậy chúng ta không có động cơ để so sánh chúng về sự giống nhau. Đó là, cặp sáu và bảy không được tạo thành một cặp ứng cử viên bởi hàm băm LSH này. Có lẽ cột sáu và bảy sẽ băm trong cùng một nhóm cho một số hàm băm khác chức năng và do đó sẽ trở thành một cặp ứng cử viên từ whoa. Nhưng từ những gì chúng ta có thể nói, chỉ nhìn vào một hàm băm này, chúng không tạo thành một cặp ứng cử viên.

Mặt khác, cột hai và sáu thực hiện băm cho cùng một nhóm. Vì vậy, hai và sáu là một cặp ứng cử viên bất kể điều gì xảy ra trong các nhóm khác. Có một cơ hội tốt là cột hai và sáu giống hệt nhau trong dải được hiển thị. Đó là những phần này của cột của họ. Là giống hệt nhau.

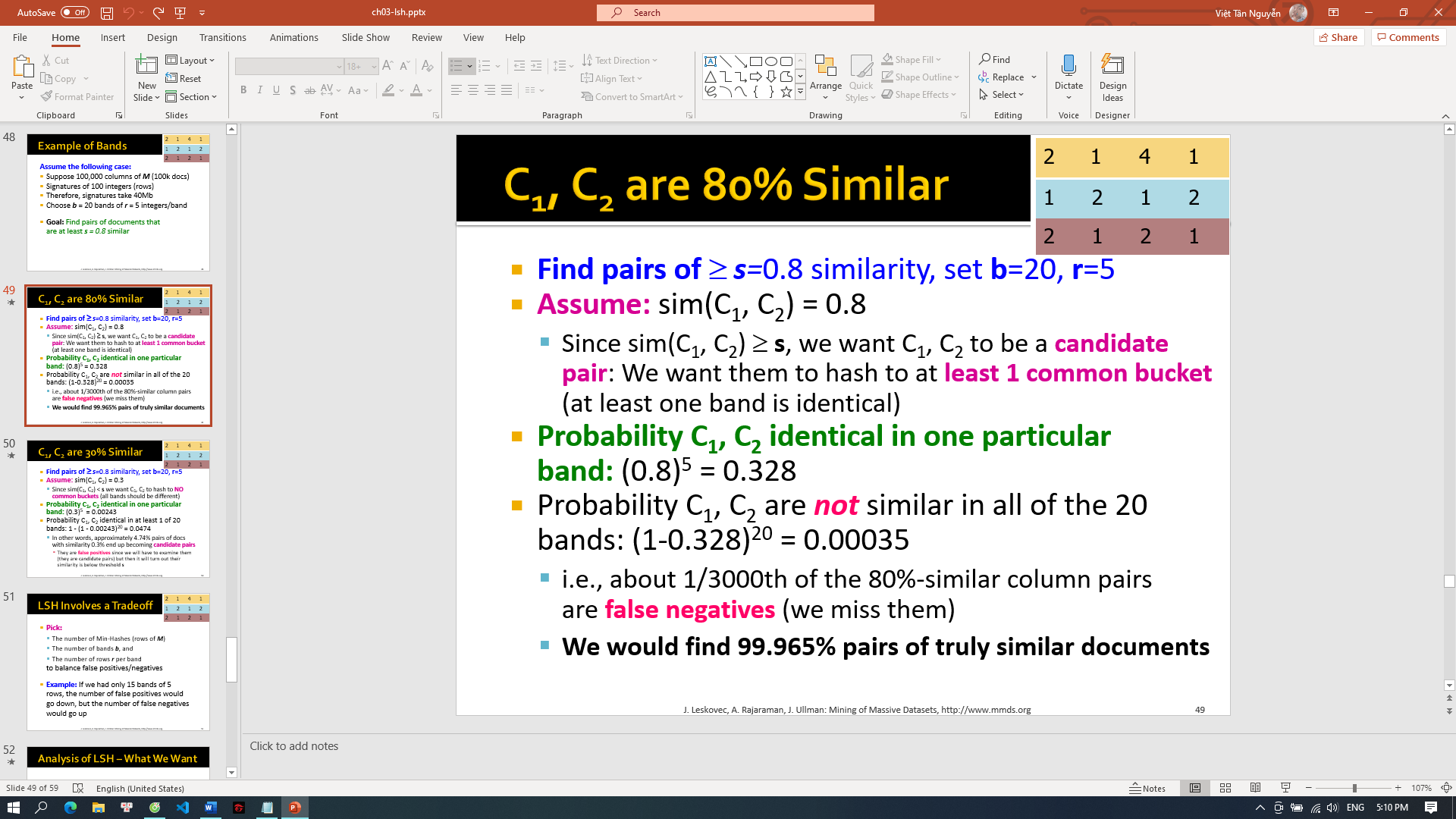
Có một cơ hội nhỏ là các phân đoạn của các cột này không giống nhau, nhưng chúng chỉ tình cờ băm cho cùng một nhóm. Nhìn chung, chúng ta sẽ bỏ qua xác suất đó vì nó có thể được tạo ra rất nhỏ, như 1 trong 4 tỷ, nếu chúng ta sử dụng nhóm công suất từ ​​2 đến 32.



Hãy xem xét một ví dụ cụ thể để có cảm nhận về xác suất dương tính giả và âm tính xảy ra như thế nào trong thực tế.

Chúng ta giả sử có 100.000 cột. Đó là, chúng ta đang tìm kiếm các tài liệu tương tự trong số 100.000 tài liệu. Chúng ta sẽ giả định rằng chữ ký có độ dài 100. Đó là, chúng ta sử dụng các hàm minhash 100 để tạo chữ ký. Do đó, ma trận chữ ký M là 100 hàng x 100.000 cột. Lưu ý rằng các chữ ký rất vừa vặn trong bộ nhớ chính. Giả sử các thành phần của chữ ký là số nguyên 4 byte, mỗi chữ ký chiếm 400 byte và tổng dung lượng yêu cầu là 40 megabyte.

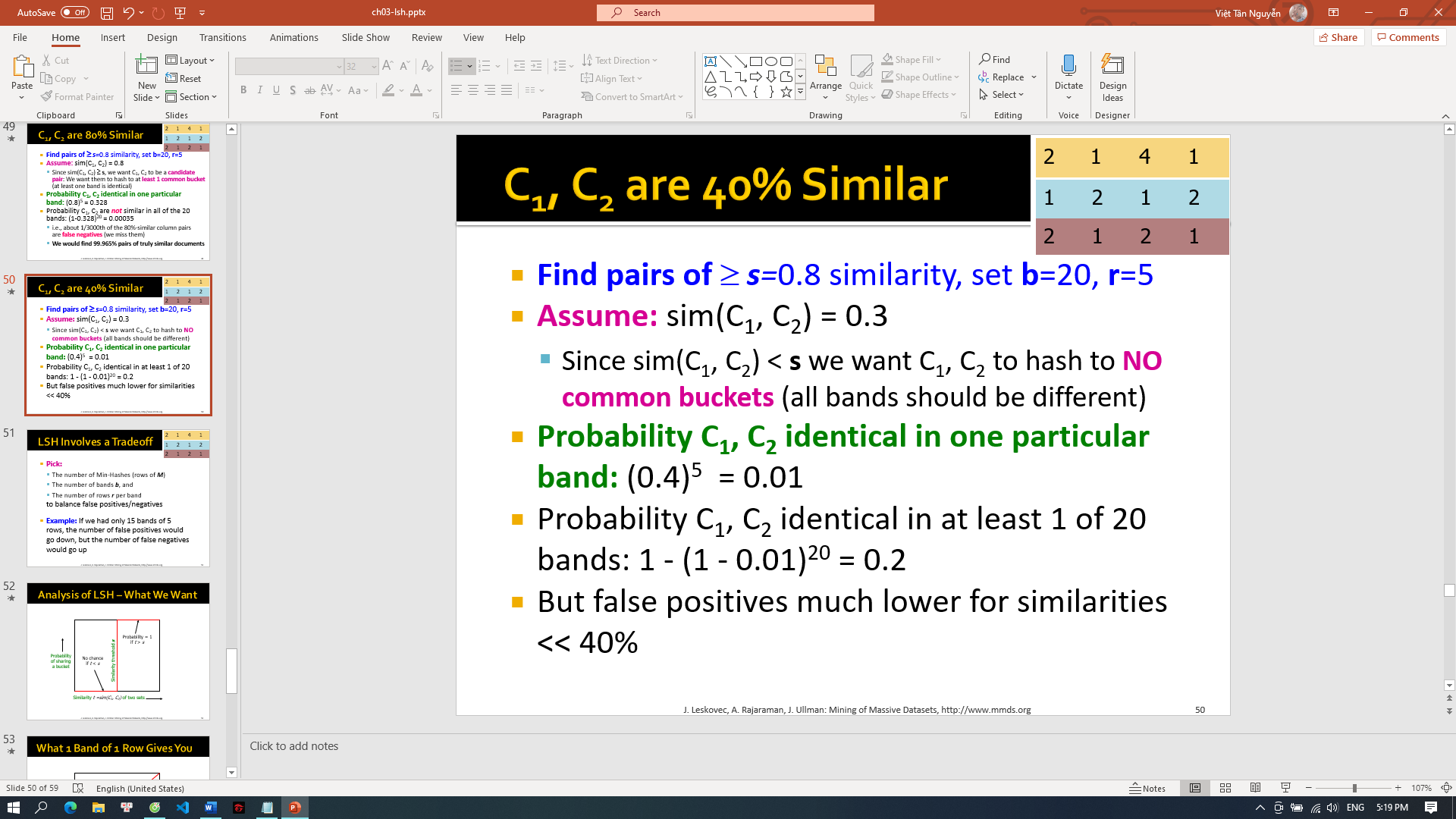
Bây giờ, hãy để ngưỡng tương tự là 80%. Đó là, chúng ta coi một cặp chữ ký tương tự nhau nếu và chỉ khi họ đồng ý ít nhất 80 trong số 100 thành phần của họ. Có khoảng 5 tỷ cặp để so sánh chúng ta muốn sử dụng LSH để tránh phải so sánh tất cả chúng. Ngẫu nhiên, nếu bạn không hiểu tại sao 5 tỷ là con số gần đúng của các cặp, số lượng chính xác các cặp mặt hàng được chọn từ 100.000 mặt hàng là 100.000 chọn hai. Đó là 100.000 nhân với 99.999 chia cho 2. Và nếu chúng ta ước tính năm số 9 với 100.000, chúng ta nhận được chính xác 5 tỷ. Trong ví dụ của chúng ta, chúng ta sẽ chia 100 hàng chữ ký của ma trận chữ ký thành 20 dải với năm hàng mỗi dải.



Đầu tiên, hãy xem xét hai cột, C1 và C2, đại diện cho các bộ có độ tương tự Jaccard 0,8. Lưu ý rằng do tính ngẫu nhiên liên quan đến băm nhỏ, các cột C1 và C2 có thể đồng ý trong nhiều hơn hoặc ít hơn 80 hàng của họ, nhưng rất có thể chúng sẽ có khoảng 80 hàng bằng nhau. Bây giờ, xác suất để các cột này giống nhau 100% là bao nhiêu một ban nhạc nhất định?

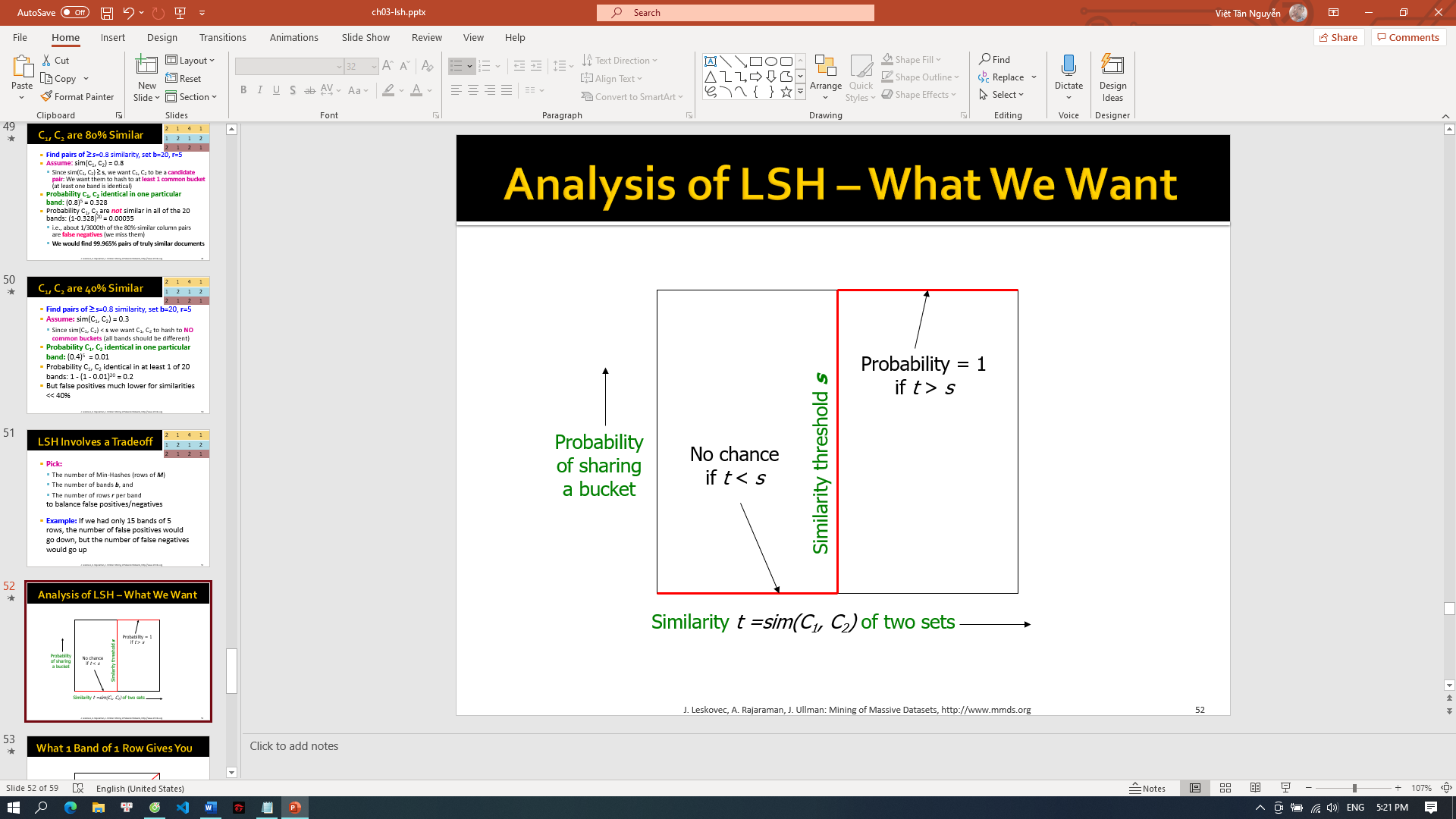
Xác suất để họ đồng ý trong một hàng bất kỳ là 0,8. Hãy nhớ rằng xác suất mà một hàm minhash đồng ý trên hai tập hợp bằng độ tương tự Jaccard của các tập hợp cơ bản. Vì vậy, xác suất để hai cột đồng ý ở tất cả năm hàng của một dải được nâng lên 0,8 lên lũy thừa thứ năm, hoặc xấp xỉ 0,328. Đó là xác suất không cao lắm, nhưng chúng ta có 20 cơ hội để biến cặp cột trở thành cặp ứng cử viên. Xác suất để chúng không băm vào cùng một nhóm trong một dải là 1 trừ 0,328 hoặc 0,672.

Nhưng xác suất các cột không thể băm thành cùng một nhóm cho bất kỳ dải nào trong số 20 dải là giá trị 0,672 được nâng lên lũy thừa thứ hai mươi, đó là một con số nhỏ. Nó là 0,00035. Cơ hội mà cặp C1 và C2 sẽ là một cặp ứng cử viên là 1 trừ đi, hoặc 0,99965. Nói một cách khác, xác suất âm tính giả, một cặp bộ có độ giống Jaccard 80%, nhưng những chữ ký không trở thành một cặp ứng cử viên là 0,00035, hoặc khoảng 1 trên 3,000.

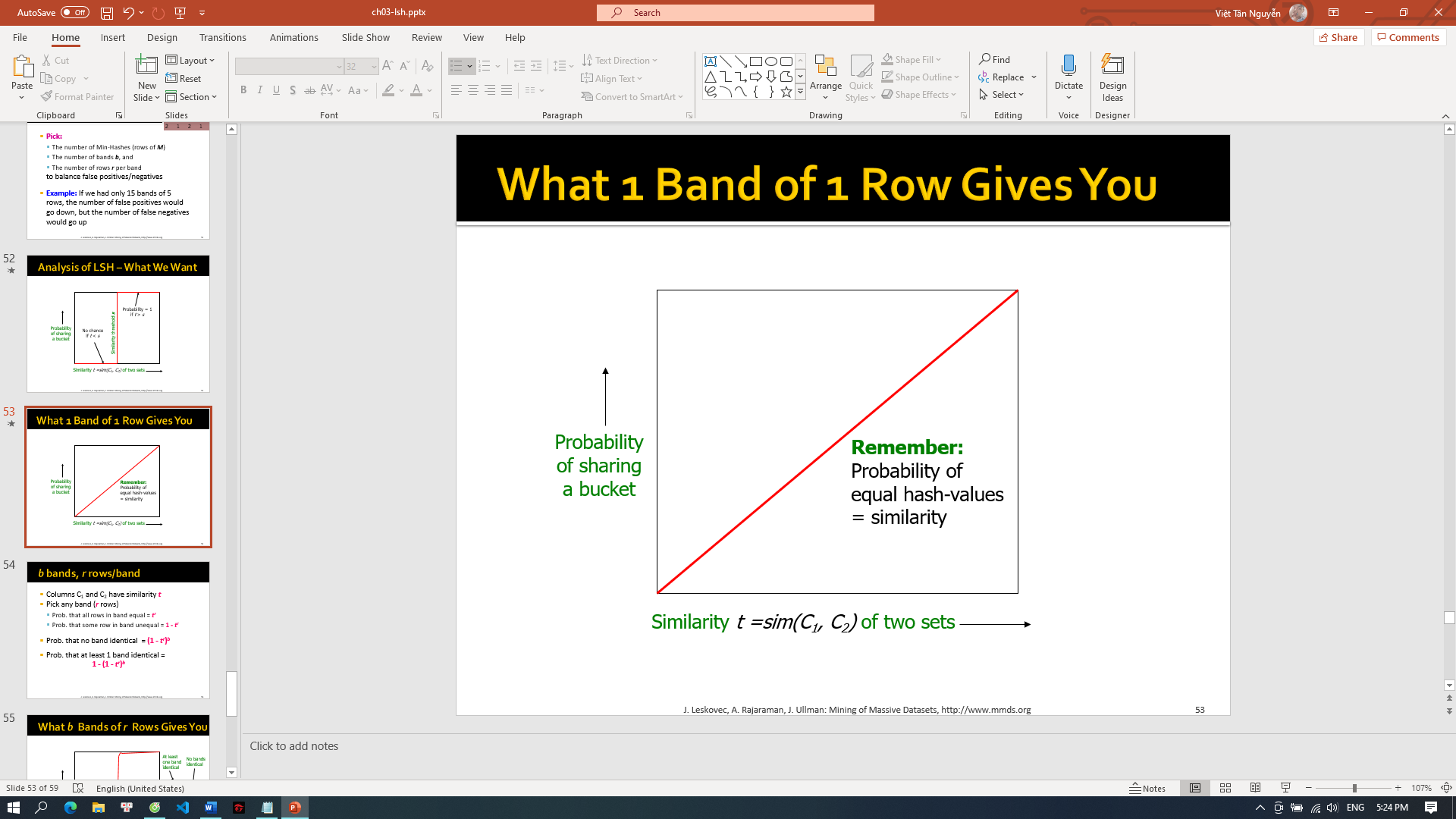


Bây giờ hãy nhìn vào một cặp bộ có độ giống nhau của Jaccard là 0,4. Xác suất các chữ ký của họ giống hệt nhau trong một dải nhất định là 0,4 đến sức mạnh thứ năm, hoặc khoảng 1%. Xác suất để chữ ký của họ băm vào cùng một nhóm trong ít nhất một trong số 20 dải chắc chắn không nhiều hơn 20 lần, hay 20%. đó, điều đó không tuyệt vời. Có nghĩa là trong số 40% tập hợp cơ bản tương tự, có 20% dương tính giả, các cặp chữ ký chúng ta sẽ phải so sánh và nhưng sẽ thấy rằng chúng không có ít nhất 80% silar, tương tự.

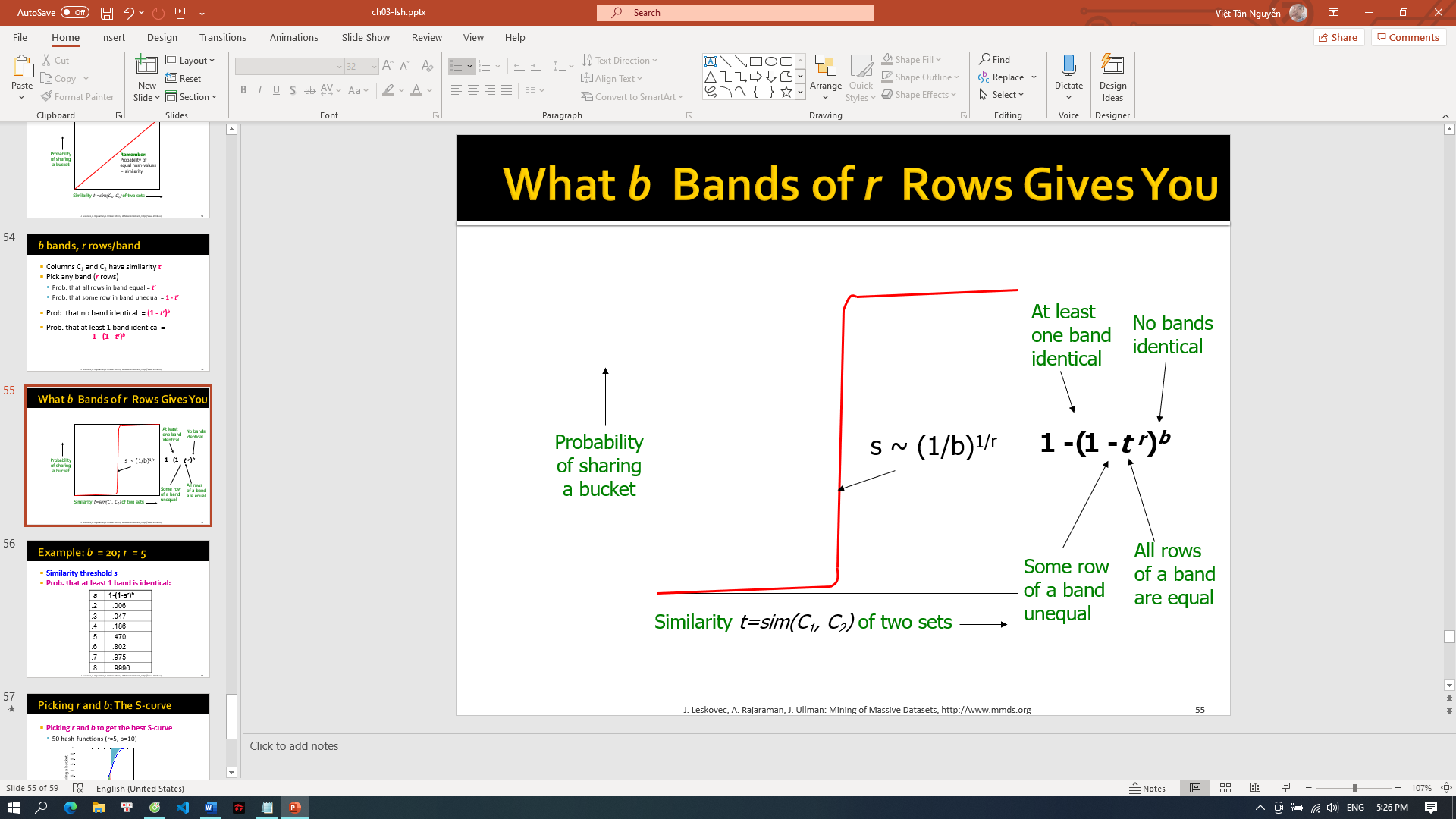
Nhưng 20% ​​dương tính giả là xấu, nhưng sai tỷ lệ tích cực giảm nhanh chóng khi mức độ giống nhau của các bộ cơ bản giảm. Ví dụ, đối với độ tương tự Jaccard 20%, chúng ta nhận được ít hơn 1% kết quả dương tính giả. Chúng ta không thể xác định chính xác số lần dương tính giả vì điều đó phụ thuộc vào sự phân bố của Jaccard Sự giống nhau giữa các tập hợp cơ bản. Ví dụ: nếu hầu hết các cặp bộ giống nhau 79%, hầu như tất cả sẽ là dương tính giả. Nhưng nếu cặp bộ điển hình có độ giống nhau về Jaccard là vài phần trăm, thì hầu như sẽ không có kết quả dương tính giả.



Một cách để xem xét vấn đề thiết kế một lược đồ LSH từ ma trận minhash là cái này. Chúng ta muốn xác suất của hai cột chia sẻ một nhóm là một hàm bước với ngưỡng t bằng giá trị mà tại đó chúng ta coi các tập cơ bản tương tự nhau. Tức là, nếu độ tương tự Jaccard của các tập cơ bản nhỏ hơn t, chúng ta muốn không có cơ hội các chữ ký sẽ chia sẻ một nhóm cho một trong các băm và do đó trở thành một cặp ứng cử viên. Tuy nhiên, nếu mức tương tự Jaccard cơ bản vượt quá 2, chúng ta muốn cặp chữ ký chắc chắn trở thành một cặp ứng cử viên.



Mặt khác, một hàng duy nhất của ma trận chữ ký cung cấp cho chúng ta điều gì? Nó cho chúng ta một đường thẳng. Sự biện minh là định lý về xác suất của hai giá trị minhash bằng độ tương tự Jaccard của tập hợp cơ bản. Điều đó không quá tệ. Ít nhất là xác suất đi đúng hướng, nhưng nó thực sự để lại rất nhiều tích cực và tiêu cực sai.

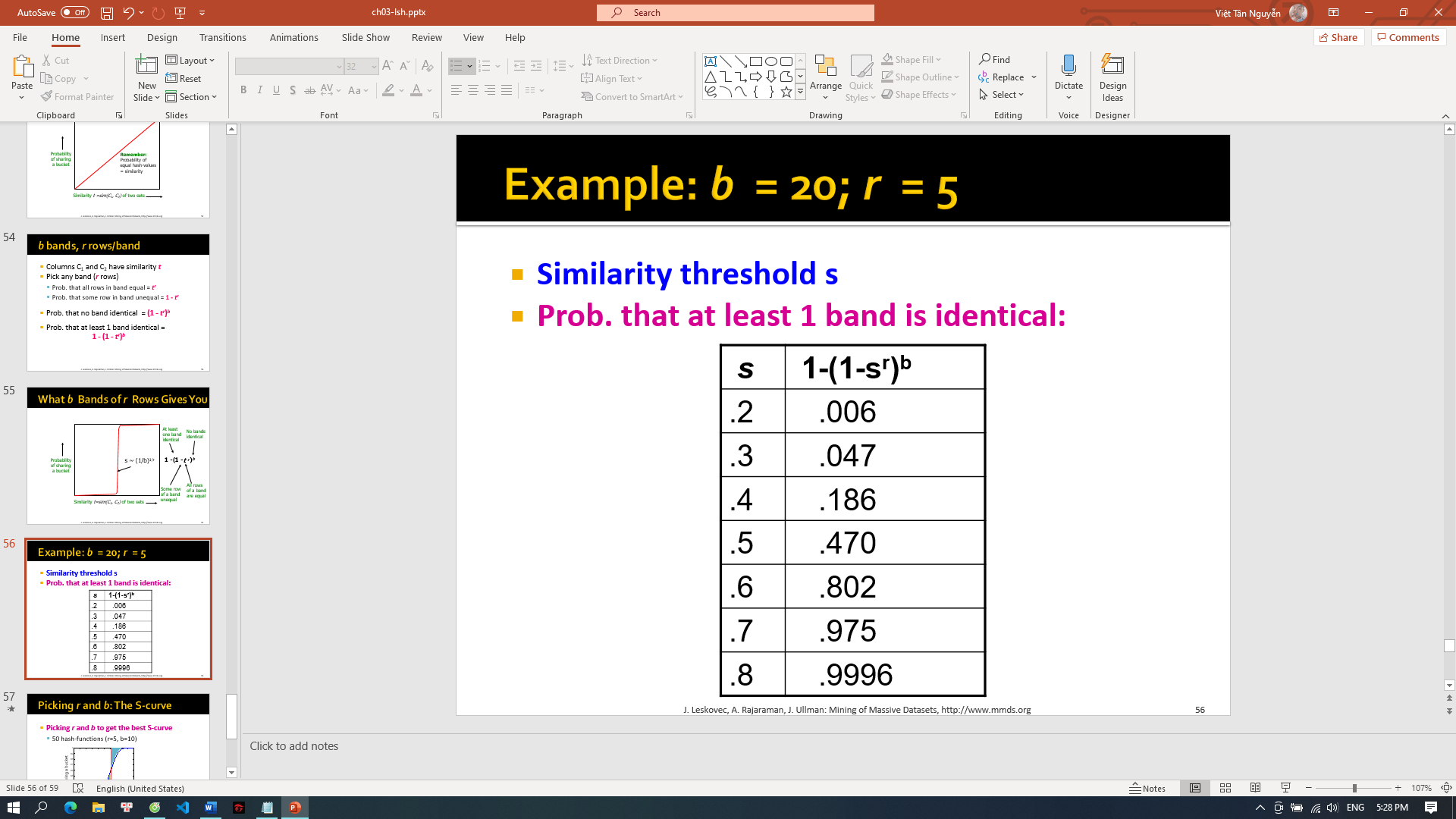


Nhưng khi chúng ta kết hợp nhiều hàm minhash thành b dải r hàng, chúng ta bắt đầu có được một hình dạng đường cong s với các vùng âm và dương sai giảm đáng kể. Chúng ta sẽ lấy hàm liên quan đến xác suất của hai tập hợp có các chữ ký trở thành một cặp ứng cử viên cho sự giống nhau của các tập hợp.

Đầu tiên, nếu các tập hợp cơ bản có sự giống nhau về Jaccard, thì xác suất chữ ký của họ sẽ giống hệt nhau trong tất cả r hàng của một dải cụ thể là s đến r. Vì vậy, xác suất để chữ ký của họ sẽ không bằng nhau trong dải này là 1 trừ s cho r.

Và xác suất mà chữ ký của họ sẽ không bằng nhau trong mỗi b ban nhạc được nâng lên sức mạnh thứ b. Cuối cùng, xác suất mà chữ ký của họ sẽ đồng ý vào lúc ít nhất một ban nhạc là một trừ đi, hoặc một trừ đi số lượng, một trừ đi s cho r đều được nâng lên lũy thừa thứ b.

Khi b và r lớn hơn, hàm này ngày càng giống với hàm bước. Và ngưỡng tại đó sự gia tăng xảy ra là khoảng 1 trên b nâng lên lũy thừa của 1 trên r. Ví dụ, trong trường hợp b bằng 20 và r bằng 5, ngưỡng sẽ là khoảng gốc thứ năm của 120, là khoảng 0,55.



Dưới đây là một số giá trị mẫu của đường cong s này cho trường hợp chúng ta đã kiểm tra, 20 dải, mỗi dải 5 hàng. Nó không hẳn là một chức năng bước, nhưng nó khá dốc ở giữa.

Ví dụ: xem xét các giá trị từ 0,4 đến 0,6. Mức tăng từ 0,4 đến 0,6 là nhiều hơn 0,6, vì vậy độ dốc trung bình của vùng này là trên 3. Mặt khác, trong vùng 0 đến 0,4, mức tăng nhỏ hơn 0,2 và vùng từ 0,6 đến 1 cũng vậy. Có nghĩa là, độ dốc nhỏ hơn một nửa ở cả hai vùng này. Nó không chính xác là một hàm bước, nhưng tốt hơn nhiều so với hàm tuyến tính mà chúng ta nhận được từ một hàng duy nhất.

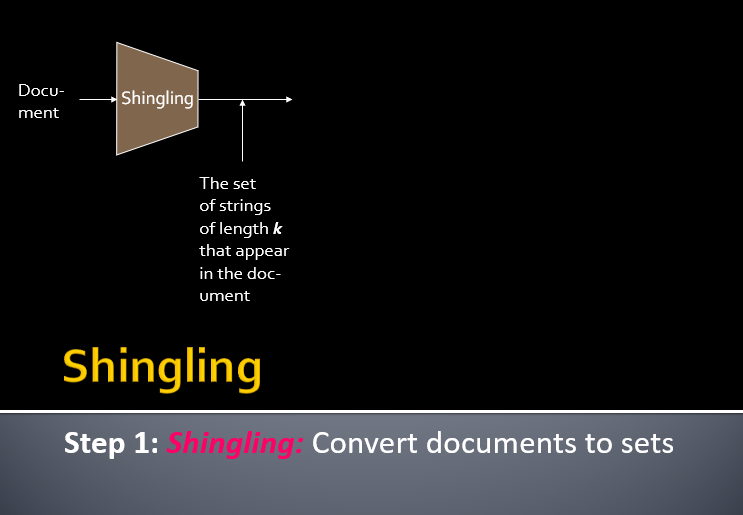
## **Summary**

Đây là tóm tắt về những gì chúng ta cần làm để tìm các nhóm có một ngưỡng t cho trước của sự tương tự Jaccard.

Đầu tiên, chúng ta cần quyết định giá trị của b và r. Như chúng ta đã đề cập, ngưỡng t sẽ xấp xỉ 1 trên b lũy thừa của 1 trên r. Nhưng có nhiều giá trị phù hợp của b và r đối với một ngưỡng nhất định. Chúng ta tạo b và r càng lớn, nghĩa là chúng ta sử dụng chữ ký càng dài, càng gần đường cong s sẽ là một hàm bước. Và do đó, chúng ta có càng ít dương tính giả và âm tính càng tốt. Nhưng chúng ta tạo chữ ký càng lâu, chúng sẽ càng chiếm nhiều không gian và công việc sẽ càng nhiều để thực hiện tất cả các thao tác băm nhỏ.

Sau đó, chúng ta phải chạy LSH để lấy các cặp ứng cử viên. Đối với mỗi cặp ứng cử viên, chúng ta kiểm tra chữ ký của họ và đếm số thành phần mà chúng đồng ý. Bằng cách đó, chúng ta có thể xác định xem liệu sự giống nhau của các chữ ký có thực sự không đạt hoặc vượt quá ngưỡng t. Chúng ta có thể dựa vào sự giống nhau của các chữ ký thực sự đo Jaccard sự giống nhau của các tập hợp cơ bản. Tuy nhiên, nếu chúng ta muốn sử dụng tài nguyên, chúng ta có thể đi đến bản thân các bộ sau khi xác định rằng chữ ký của chúng hoàn toàn giống nhau.

Trong một số trường hợp, sự giống nhau của các sản phẩm dinh dưỡng sớm con sẽ đánh giá quá cao sự tương đồng của các tập hợp mà chúng đại diện, vì vậy có thể là hai bộ không thực sự đủ giống nhau. Bằng cách tính toán sự giống nhau về Jaccard của các tập hợp cơ bản, chúng ta có thể loại bỏ các kết quả dương tính giả. Thật không may, chúng ta không thể loại bỏ các phủ định sai theo cách này. Nếu hai tập hợp có sự giống nhau về Jaccard trên ngưỡng, nhưng không may mắn thay, chữ ký của họ không bao giờ trở thành một cặp ứng cử viên, thì chúng ta sẽ không bao giờ xem xét cặp chữ ký này hoặc bộ cơ bản của chúng.



**STEP 1: SHINGLING: CONVERT DOCUMENTS TO SETS**

**TÀI LIỆU DƯỚI DẠNG DỮ LIỆU HIGH-DIM**

Các cách tiếp cận đơn giản:

* Tài liệu = tập hợp các từ xuất hiện trong tài liệu
* Tài liệu = tập hợp các từ "quan trọng"
* Không hoạt động tốt cho ứng dụng này. Tại sao?

Chú ý: Cần phải tính đến thứ tự của các từ!

Một cách khác: **Shingles!**

**ĐỊNH NGHĨA: SHINGLES**

K-shingle (hoặc k-gram) cho 1 tài liệu là một chuỗi các tokens k xuất hiện trong tài liệu

* Token có thể là ký tự, từ ngữ hoặc thứ gì đó khác, tùy thuộc vào ứng dụng
* Giả sử tokens = ký tự cho ví dụ

**Ví dụ:** k =2; tài liệu D1 = abcab  
Tập hợp 2-shingles: S(D1) = {ab, bc, ca}

Tùy chọn: shingles dưới dạng 1 cái túi (nhiều bộ), đếm ab hai lần: S'(D1) = {ab, bc, ca, ab}

**NÉN SHINGLES**

Để nén shingles dài, chúng ta có thể băm chúng thành 4 bytes

Đại diện cho một tài liệu bằng tập các giá trị băm của k-shingles

* Ý tưởng: Hai tài liệu có thể (hiếm khi) có shingles chung, trong khi trên thực tế chỉ có các giá trị băm được chia sẻ

**Ví dụ:** k=2; tài liệu D1= abcab

Tập của 2-shingles: S(D1) = {ab, bc, ca}

Băm singles: h(D1) = {1, 5, 7}

**METRIC TƯƠNG TỰ CHO SHINGLES**

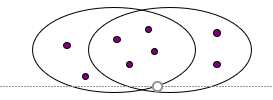
Tài liệu D1 là một tập các k-shingles của nó C1= S(D1)

Tương đương, mỗi tài liệu là một vectơ 0/1 trong không gian k-shingles

* Với shingle độc nhất là một chiều không gian
* Các vectơ rất thưa thớt

**Một biện pháp tự nhiên tương tự là sự tương đồng Jaccard:**

sim(D1,D2) -| C1∩C2|/| C1∪C2 |



**GIẢ ĐỊNH CÔNG VIỆC**

Các Tài liệu có nhiều shingles chung thì có văn bản tương tự, ngay cả khi thứ tự văn bản xuất hiện theo cách khác nhau

Cảnh báo: Nếu bạn phải chọn **k** đủ lớn, hoặc hầu hết các tài liệu sẽ có shingles

* k = 5 là OK “ổn” cho các tài liệu ngắn
* k = 10 là better “tốt hơn” cho các tài liệu dài

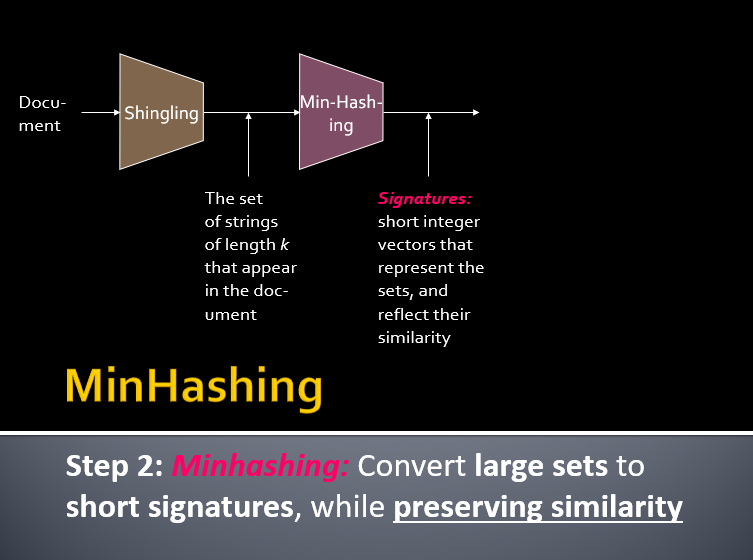
**ĐỘNG LỰC CHO MINHASH / LSH**

Giả sử chúng ta cần tìm các tài liệu gần trùng lặp trong số hàng triệu tài liệu

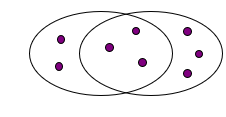
Chúng ta sẽ phải tính toán pairwise Jaccard similarities (các cặp tương đồng Jaccard)cho mỗi cặp tài liệu

* ≈ 5\*1011 so sánh
* Với tốc độ 105 giây/ngày và 106 so sánh/giây, nó sẽ mất **5 ngày**

Đối với **N = 10** hàng triệu, phải mất hơn một năm



**MÃ HÓA SETS DƯỚI DẠNG BIT VECTORS**

****Nhiều vấn đề tương tự có thể được chính thức hóa như tìm các tập hợp con có giao điểm quan trọng

Bộ mã hóa sử dụng vectơ 0/1 (bit, boolean)

* Một kích thước trên mỗi phần tử trong tập hợp phổ quát

Chứng minh đặt giao điểm (intersection) theo bitwise AND và đặt phép hợp (union) theo bitwise OR

**Ví dụ:** C1 = 10111; C2 = 10011

* Kích thước intersection = 3; kích thước của union = 4,
* Jaccard similarity (không phải khoảng cách) = 3/4
* Khoảng cách: d(C1,C2) = 1 – (Jaccard similarity) = 1/4

**TỪ SETS ĐẾN BOOLEAN MATRICES**

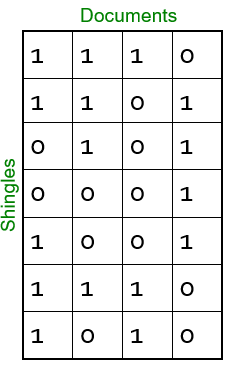
Hàng = phần tử (shingles)

Cột = tập hợp (tài liệu)

* 1 trong hàng e và cột s nếu và chỉ khi e là thành viên của s
* Tương tự cột là Jaccard similarity của các tập hợp tương ứng (hàng có giá trị 1)
* Ma trận điển hình là ma trận thưa thớt!

**Ví dụ:** sim(C1, C2) = ?

* Kích thước intersection = 3; kích thước của union = 6, Jaccard similarity (không phải khoảng cách) = 3/6
* d(C1,C2) =1 – (Jaccard similarity) = 3/6



**OUTLINE: TÌM CÁC CỘT TƯƠNG TỰ**

* Tài liệu → Tập hợp cỉa shingles
* Đại diện cho các tập hợp dưới dạng vectơ boolean trong ma trận
* Tìm các cột tương tự trong khi tính toán chữ ký nhỏ
* Sự giống nhau của cột == sự giống nhau của chữ ký
* Tìm các cột tương tự, Chữ ký nhỏ

Cách tiếp cận **Naive**:

1) Chữ ký của các cột: tóm tắt nhỏ các cột

2) Kiểm tra các cặp chữ ký để tìm các cột tương tự

*Điều cần thiết*: Điểm tương đồng của chữ ký và cột có liên quan

3) Tùy chọn: Kiểm tra xem các cột có chữ ký tương tự có thực sự giống nhau không

Cảnh báo:

So sánh tất cả các cặp có thể tốn quá nhiều thời gian: Công việc cho LSH

* Các phương pháp này có thể tạo ra phủ định sai và thậm chí là khẳng định sai (nếu kiểm tra tùy chọn không được thực hiện)

**CỘT BĂM (CHỮ KÝ)**: "băm" mỗi cột **C** đến một **chữ ký nhỏ** **h(C),** sao cho:

* h(C) đủ nhỏ để chữ ký phù hợp với RAM
* sim(C1, C2) giống như "sự tương đồng" của chữ ký h(C1) và h(C2)

Mục tiêu: Tìm hàm băm h(·) như vậy:

* Nếu sim (C1,C2) cao, thì với prob cao**. h(C1) = h(C2)**
* Nếu sim (C1,C2) thấp, thì với prob cao. **h(C1) ≠ h(C2)**

Băm docs vào buckets. Mong đợi rằng "hầu hết" các cặp tài liệu gần trùng lặp băm vào cùng một bucket!

**MIN-HASHING** : Tìm hàm băm h() như vậy:

* Nếu sim (C1,C2) cao, thì với prob cao. **h(C1) = h(C2)**
* Nếu sim (C1,C2) thấp, thì với prob cao**. h(C1) ≠ h(C2)**

Rõ ràng, hàm băm phụ thuộc vào metric tương tự:

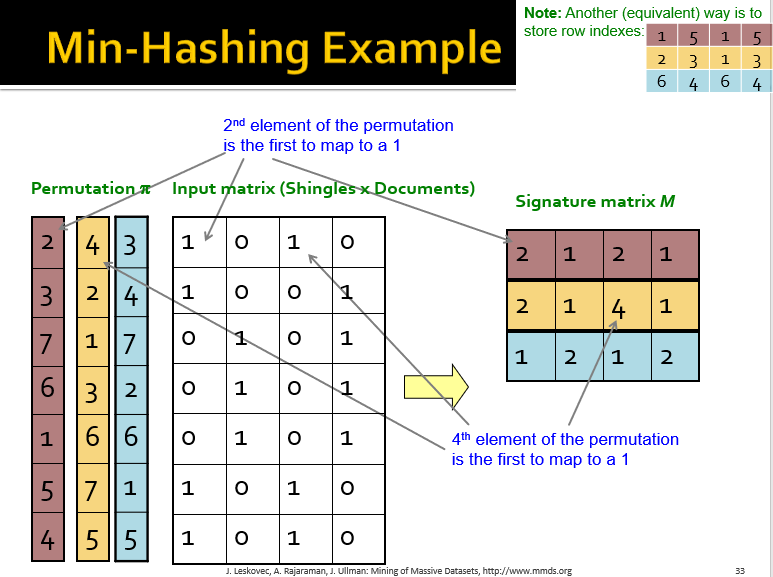
Không phải tất cả các chỉ số tương tự đều có hàm băm phù hợp

Có một hàm băm phù hợp cho Jaccard similarity: Nó được gọi là Min-Hashing

Hãy tưởng tượng các hàng của ma trận boolean được hoán vị dưới dạng hoán vị ngẫu nhiên π

Xác định hàm "băm" hπ(C) = chỉ mục của hàng đầu tiên (theo thứ tự hoán vị π) trong đó cột C có giá trị là 1: hπ(C) = minππ(C)

Sử dụng một số hàm băm độc lập (nghĩa là hoán vị) để tạo chữ ký của một cột



Chọn hoán vị ngẫu nhiên π

Yêu cầu bồi thường: Pr[hπ(C1)- hπ(C2)]- sim(C1,C2)

Hãy để X là một doc (set of shingles), y X là một∈ shingle

Sau đó: Pr[(y) = min((X))] = 1/| X|ππ

Cũng có khả năng bất kỳ y X nào ∈ được ánh xạ đến phần tử min

Hãy để y được s.t π (y) = min(π (Cππ1∪C2))

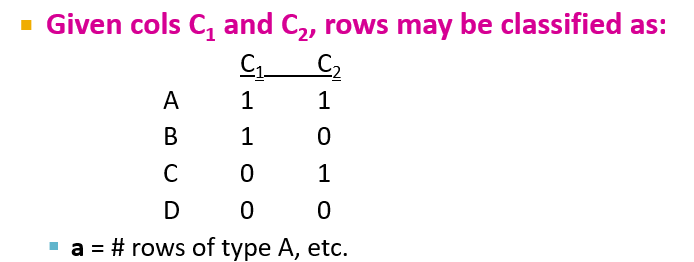
Sau đó:

π (y) = min(π (C1))nếu y C∈1 hoặc

π (y) s min (π (C2))nếu và C∈2

Vì vậy, prob. rằng cả hai đều đúng là prob. y ∈ C1 ∩ C2

Pr[min(π(C1))=min(π(C2))]=|C1∩C2|/|C1∪C2|= *sim*(C1, C2)



**Ghi chú :** sim(C1,C2) =a/(a+b+c)

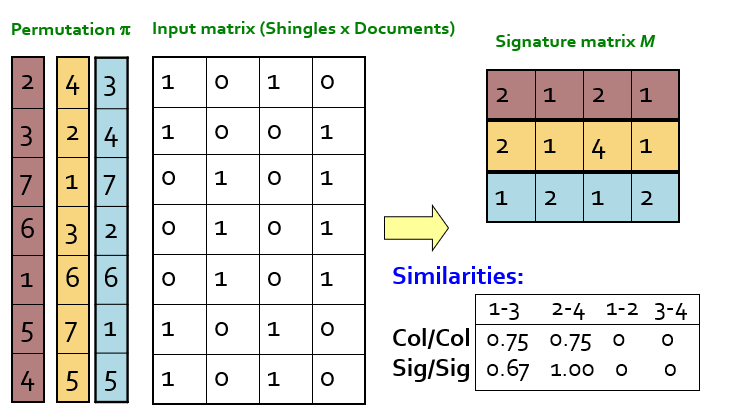
**Sau đó :** Pr[*h*(C1) = *h*(C2)] = *Sim*(C1, C2)

* + Nhìn xuống cols C1 và C2 cho đến khi chúng ta thấy A1
  + Nếu đó là hàng loại A thì h(C1)=h(C2)
  + Nếu là hàng loại B hoặc loại C thì không

Chúng ta có : Pr[*h*π(C1) = *h*π(C2)] = *sim*(C1, C2)

* Sau đó chúng ta tổng quát hóa thành nhiều hàm băm
* Sự giống nhau của hai chữ ký là phần nhỏ của các hàm băm mà chúng đồng ý

**Ghi chú** : Do thuộc tính Min-Hash, sự giống nhau của các cột cũng giống như sự giống nhau mong đợi của các chữ ký của chúng.



Chọn K = 100 hoán vị ngẫu nhiên của các hàng

* Sig (C) như một vectơ cột
* sig (C) [i] = theo hoán vị thứ i, chỉ số của hàng đầu tiên có số 1 trong cột C

***sig*(C)[i] = min (πi(C))**

*Ghi chú : Bản phác thảo (chữ ký) của tài liệu C có kích thước nhỏ gần bằng 100 Byte*