دانشکده علوم ریاضی و آمار

**مدرس: دکتر مجتبی رفیعی** نیمسال اول ۱۴۰۰-۱۴۰۱

ساختمان داده‌ها و الگوریتم‌ها

# جلسه ۴۲

نگارنده: فاطمه کریمیان

۱۰ دی ۱۴۰۰

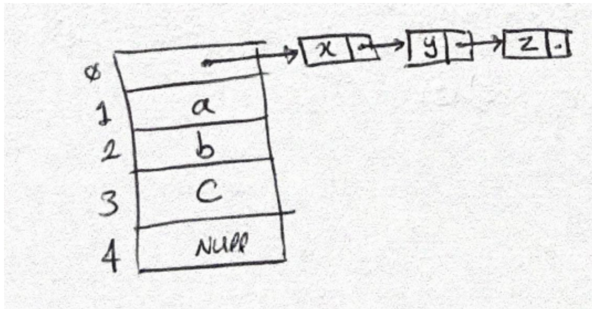
## فهرست مطالب



## ۱ درهم سازی سراسری (Universal Hashing)

در تنظیمات مربوط به جداول درهم سازی با قید حافظه که در آنها $n >> m$ است، یک شخص متخاصم می‌تواند $n$ کلید را طوری انتخاب کند و در ساختار ذخیره نماید که هزینه بازیابی آن از مرتبه $O(n)$باشد. این مشکل بدین دلیل ایجاد می‌شود که تابع درهم ساز مربوط به جدول درهم ساز $HT$ همواره ثابت است. ویژگی مذکور در برخی از کاربردهای عملی مثل کامپایلرها که از جدول درهم ساز برای ذخیره و بازیابی متغیرهای یک برنامه بهره می‌گیرند، ضروری است و سبب افزایش چشمگیری در کارایی می‌شود.

**مثال:** فرض کنید برنامه ما شامل شش متغیر $x, y, z, a, b, c$ است و جدول درهم سازی $HT$ نیز شامل ۵ خانه است و متغیرها توسط تابع درهم $h$ از این جدول $HT$ به صورت زیر نگاشته شده‌اند.




فرض کنید برنامه ما هم به نحوی است که اکثرا فقط از متغیرهای $x, y, z$ استفاده می‌کنیم. با این اوصاف واضح است که در چنین تنظیمی همواره دسترسی به متغییرهای $a, b, c$ هزینه کمتری نیاز دارد و چنین تنظیمی برای هر بار کامپایل ثابت است.
اگر برای هر بار کامپایل، درهم سازی متفاوتی صورت گیرد، می‌تواند نویدی برای هزینه کمتر کامپایل فارغ از برنامه و متغیرهای آن باشد.
**جمع‌بندی:** با توجه به مباحث بالا در ادامه سعی می‌کنیم به سمت معرفی یک خانواده متناهی از توابع درهم ساز حرکت کنیم که برای ساخت جدول درهم ساز $HT$ به طور تصادفی یکی از توابع داخل آن را انتخاب و درهم سازی را براساس آن انجام می‌دهیم. لازم به ذکر است که به محض انتخاب تابع درهم ساز، تابع در حین اجرا ثابت باقی می‌ماند ولی تا قبل از اجرا مشخص نیست.
**نکته:** درهم سازی سراسری که ایده بالا را در دل خود جای داده است برای هر برنامه کاربردی (Application) مناسب نیست و باید در انتخاب آن دقت کرد. به عنوان مثال، یک برنامه کاربردی که از یک پایگاه داده ثابت طبق یک قرارداد مشخصی استفاده میکند گزینه مناسبی برای بکارگیری درهم سازی سراسری نیست چرا که ممکن است منجر به ایجاد مخاطره در صحت و درستی برنامه کاربردی گردد.

## ۲ تنظیمات رویکرد درهم سازی سراسری

اندیس خانه‌های حافظه $= [m] = \{0, 1, 2, ..., m-2\}$
دامنه کلیدها $= U$
خانواده متناهی از توابع درهم ساز $= H = \{h_1, ..., h_l\},\ where\ h_i : U \to [m]$
$m << n \leq |U|$

جائیکه $n$ بیانگر تعداد عناصر مجموعه پویا (یا به طور دقیق‌تر تعداد عناصر فعلی جدول درهم ساز $HT$ )می‌باشد.

**سوال:** مجموعه توابع درهم ساز $H$ چه ویژگی یا شرایطی باید داشته باشد؟
**خصیصه** $Universal$: مجموعه H دارای ویژگی $Universal$ است اگر به ازای هر زوج $k, l \in U$ تعداد توابع درهم ساز $h \in H$ که $h(l) = h(k)$ است، حداکثر $\frac{|H|}{m}$ باشد. به عبارت دیگر با انتخاب یک تابع درهم ساز تصادفی از H احتمال بروز برخورد در $h(l) = h(k)$ بیشتر از $\frac{1}{m}$ نیست. به زبان ساده‌تر یعنی $l$و $Kk$در $hash$های مختلف عملا در خانه‌های مختلف توزیع می‌شوند.
**مثال-مجموعه توابع درهم ساز**$H$: فرض کنید p یک عدد اول بزرگتر از همه اعداد داخل U باشد.

$Z_p^* = \{1, ..., p-1\}$
$Z_p = \{0, ..., p-1\}$

تابع درهم ساز $h_{a,b}$ را به صورت زیر تعریف می‌کنیم:

$$h_{a,b}(k) = \big((ak + b)\ mod\ p\big)\ mod\ m$$

جاییکه $a \in z_p^*$ و $b \in z_p$ می‌یاشد. حال مجموعه توابع درهم ساز H را به صورت زیر تعریف می‌کنیم:

$$h_{p,m} = \{h_{a,b}\ : a \in z_p^*,\ b \in z_p\}$$

با توجه به مقادیر کاندید برای a وb پر واضح است که:

$$|h_{p,m}| = p(p-1)$$

در فصل 11 کتاب مرجع در قضیه 11.5 صفحه ۲۶۷ خصیصه Universal بودن برای خانواده فوق نشان داده شده است که علاقمندان برای مطالعه بیشتر می‌توانند به آن مراجعه کنند.

**قضیه:** اگر مجموعه درهم ساز H (تشریح شده در بالا) دارای خصیصه Universal باشد، آنگاه به صورت میانگین هزینه درج / حذف/ جستجو از مرتبه $O(1+\alpha)$ می‌باشد.

**نکته:** در رویکرد درهم سازی سراسری، ابتدا تنظیمات مربوطه به جدول درهم ساز انجام می‌شود و سپس در هر بار اجرای آن یک تابع $h_i$ از مجموعه درهم ساز $H$به صورت تصادفی انتخاب می‌شود.

**نکته:** قبلا دیدیم که در ارائه یک جدول درهم ساز برای تنظیماتی که در آن $m << n$ است، متوازن بودن تابع درهم ساز بیانگر یک تابع خوب برای جدول درهم ساز به شمار می‌آمد. با اینحال بیان یک فرمول کلی نمی‌تواند بیانگر یک تابع درهم ساز خوب برای هر کاربری باشد. برای این منظور، به مثال زیر دقت کنید.

**مثال:** یک جدول درهم ساز با تنظیمات زیر را در نظر بگیرید:

اندازه حافظه $= [m] = \{0, 1, 2, ..., 999\}$
دامنه $= U = [10^8]$
تابع درهم ساز $= h(k) = k \ mod \ 1000$
$m << n$

فرض کنید جدول درهم ساز فوق می‌خواهد در دانشگاه اصفهان به کار گرفته شود و شماره دانشجویی ما در این دانشگاه به شکلی است که بر 1000 تقسیم‌پذیر است. بنابراین، با جدول درهم ساز فوق، همه دانشجویان به خانه 0 نگاشت می‌شوند و 999 خانه حافظه خالی می‌مانند.

**جمع‌بندی:** جداول درهم ساز می‌بایست متناسب با کاربرد طراحی شده و نمی‌توان از یک فرمول کلی بهره گرفت.

## ۳ اهمیت اندازه حافظه

همان طور که بیشتر نیز تاکید کردیم، هدف ما داشتن یک تابع درهم ساز متوازن (خوب) است. در این راستا، تعیین مقدار مناسب حافظه می‌تواند تاثیر گذار باشد. به مثال زیر دقت کنید.

**مثال:** اگر تابع درهم ساز زیر را انتخاب کرده باشیم، باید به این مورد دقت کنیم که m ای که لحاظ می‌کنیم نباید توان ۲ باشد ($m =\neq 2^\alpha$).

$h(k) = k \ mod \ m$

فرض کنید کلید $k$ دارای $l$-bit باشد.

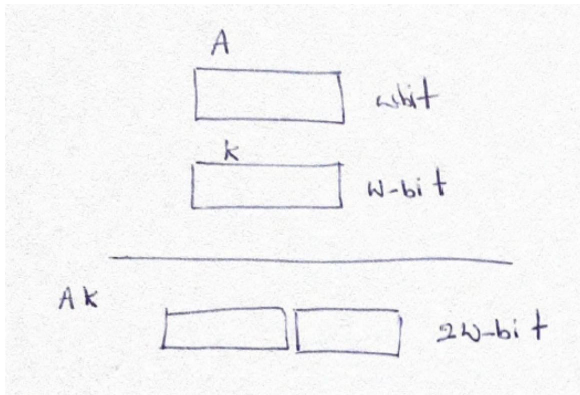
$$k = (a_{l-1} \ q_{l-2} \ ...a_{\alpha+2} \ a_{\alpha+1} \ \ a_\alpha \ ...a_2 \ a_1 \ a_0)_b$$
$$= a_0 \ * 2^0 \ + a_1 \ * 2^1 + ... + a_\alpha \ * 2^\alpha + a_\alpha + 1 \ * 2^\alpha + 1 + a_\alpha + 2 \ * 2^\alpha + 2 + ... + a_{l-1} \ * 2^{l-1}$$

واضح است که از آنجاییکه محاسبه در پیمانه $2^\alpha$ انجام می‌شود جملاتی که ضریب $2^\alpha$ دارند عملی تاثیری در نتیجه ندارند. بنابراین، تنها $\alpha$ بیت کم ارزش k در خروجی تابع درهم ساز $h$ تاثیر گذار است .
**نکته:** تجربه نشان می‌دهد که برای مثال‌های مشابه مثال فوق که از پیمانه استفاده می‌کنند، انتخاب یک عدد اول برای $m$ می‌تواند گزینه مناسبی باشد. به مثال معرفی شده در بخش‌های قبل که مربوط به تابع درهم ساز بد بود، مراجعه کنید. یک شهود برای توجیه این انتخاب می‌تواند این باشد که سبب می‌شود توزیع دامنه بین خانه‌های حافظه به صورت مناسبی انجام شود.
در ادامه تابع درهم سازی را معرفی می‌کنیم که برخلاف مثال قبل، اندازه حافظه$m = 2^\alpha$برای آن مناسب است.

**مثال:**

$h(k) = \lfloor m(kA \ mod \ 1) \rfloor$

لازم به ذکر است که در عبارت بالا، نوع mod خاص مدنظر است که تنها قسمت اعشار را نگه می‌دارد. جاییکه $0 < A < 1$ و توصیه شده که $A = \frac{\sqrt{5}-1}{2}$ و $m = 2^{\alpha}$ به سبب اینکه تمام بیت‌های $k$ را در محاسبه تابع $h$دخیل می‌کند، پارامترهای مناسبی برای تابع فوق هستند. در ادامه سعی داریم تا شهود کافی برای مناسب بودن $m = 2^{\alpha}$ را بیان کنیم.
می‌دانیم که $(Ak \; mod \; 1)$ در واقع قسمت اعشار حاصلضرب $Ak$ را در خود جای می‌دهد. فرض کنید که $A$ و $k$ هر کدام $w - bit$ای می‌باشند.




طبق تابع درهم‌ساز بالا، اگر $m = 2^{\alpha}$ را در Ak ضرب کنیم، آنگاه $\alpha$ بیت از Ak که بیانگر یک کسر است به سمت چپ شیف داده شده و به عنوان خروجی تابع h در نظر گرفته می‌شود. واضح است که برای این $\alpha$ بیت پرارزش حاصلضرب Ak، تمام بیت‌های $A$ و $k$ در محاسبه $\alpha$ نقش ایفا کرده‌اند. یک مثال شهودی برای ایفای نقش بیت‌ها در ادامه آرده شده است.