دانشکده علوم ریاضی و آمار

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

ساختمان داده‌ها و الگوریتم‌ها

جلسه ۴۰

نگارنده: فرشته قادری

۱۴ دی ۱۴۰۰

فهرست مطالب

۱ داده ساختار جدول درهم ساز

یاد آوری: هدف از معرفی و بررسی داده ساختارها، ذخیره و بازیابی مجموعه‌های پویا به صورت کارا می باشد. در این راستا داده ساختارهای زیادی معرفی شده‌اند و بر روی آنها عملیات مختلفی متناسب با کاربرد مورد نیاز تعریف شده‌اند. در حالت کلی داده ساختارهایی که تا به حال فرا گرفتیم در دو رده کلی زیر قابل تقسیم بندی است:

- **داده ساختارهای مقدماتی، نظیر**: لیست، لیست پیوندی صف، صف حلقوی و پشته.
- **داده ساختار های پیشرفته نظیر**: درخت trie، درخت دودویی عبارت BET، درخت دودویی جست و جو و درخت هرم بیشینه.

سوال: پیچیدگی زمانی عملیات پایه روی داده ساختارهای فوق نظیر: عملیات درج، حذف و جست و جو را قبلا بررسی کردیم، در اینجا سوال آن است که آیا می‌توان کلیه این عملیات پایه را در $O(1)$ انجام داد، یا سعی کرد تا جایی که ممکن است، حداقل مرتبه پیچیدگی زمانی را برای

این عملیات داشت؟

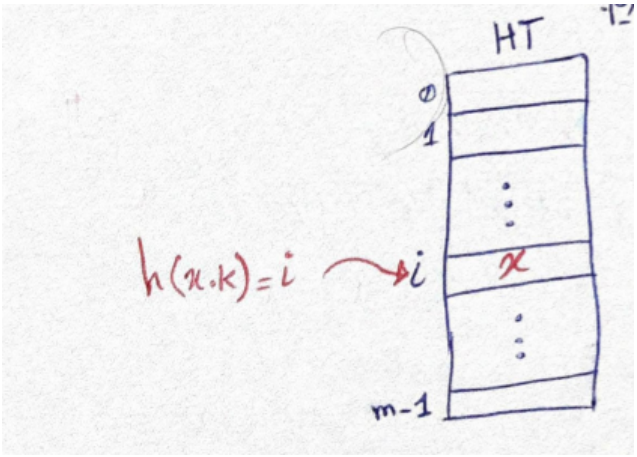
در ادامه به بررسی داده ساختار جدول درهم‌ساز برای پاسخگویی به این سوال می‌پردازیم.

تعریف داده ساختار جدول درهم‌ساز. فرض کنید:

- m خانه از حافظه را در اختیار داریم که از صفر تا m-۱ اندیس گذاری شده است.
- n کلید که مقادیر آن از صفر تا n-۱ است نیز در اختیار داریم
- همچنین یک تابع در هم ساز h که با توصیف زیر نیز داده شده است:

$$h : [0, ..., n-1] \rightarrow [0, ..., m-1]$$

یک جدول درهم‌ساز HT با m خانه حافظه، داده ساختاری است که هر کلید k عوض مجموعه $\{0, \cdots, n-1\}$ را با استفاده از تابع درهم‌ساز h به طور مناسبی به یکی از m خانه حافظه نگاشت کرده و عنصر حاوی کلید k را در آن خانه از حافظه درج کند. شکل گرافیکی یک جدول درهم‌ساز در ادامه آورده شده است. با فرض اینکه x یک عنصر است و h(x،k)=i داریم:

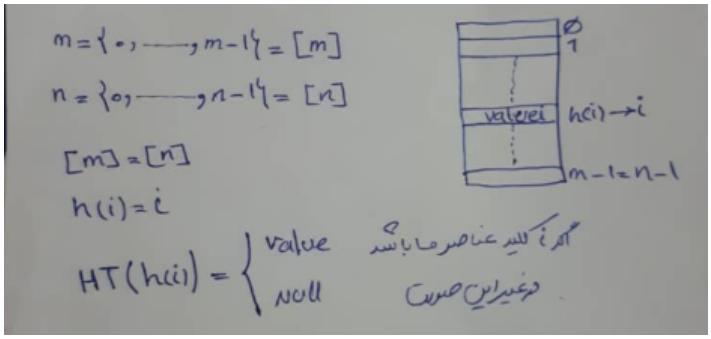


در ادامه سعی داریم به چالش های اصلی داده ساختار جدول درهم ساز، یعنی:

- تعیین خانه های حافظه،
- تعیین یک تابع درهم‌ساز خوب بپردازیم.

۲ دسترسی مستقیم یا آدرس دهی مستقیم

در این بخش به بررسی یک حالت ایده‌آل از نظر کارایی برای جداول درهم‌ساز می‌پردازیم. در این روش،با فرض عدم محدودیت روی حافظه می‌توان تنظیمات زیر را برای جدول درهم‌ساز درنظر گرفت:



در این روش، عملیات درج، حذف و جستجو به صورت زیر قابل اعمال است:

- درج مقدار value برای عنصر با کلید بهk صورت $HT[h(k)] = HT[k] = value$ انجام می‌شود.
- حذف عنصر با کلید k به صورت $HT[h(k)] = HT[k] = Null$ انجام می‌شود.
- جستجو یک عنصر با کلید k به صورت زیر قابل انجام است:

- اگر $HT[h(k)] = HT[k] = Null$ باشد، آنگاه عنصر با کلید k در مجموعه پویای خود نداریم.
- اگر $HT[h(k)] = HT[k] \neq Null$ باشد، عنصر با کلید k موجود است و می‌توانیم آن عنصر را بازیابی کنیم.

نکته: روش آدرس دهی مستقیم،یک راه حل نظری و نه علمی است چرا که فضای مصرفی زیادی را می‌طلبد. با این حال پیچیدگی زمانی عملیات پایه: درج، حذف و جستجو در آن از مرتبه $O(1)$ است.

۳ حرکت به سوی جداول درهم‌ساز عملی (با قید محدودیت حافظه)

در ادامه به دنبال راه حلی هستیم که با اعمال محدودیت حافظه بتوانیم داده ساختار جدول درهم ساز تا حد امکان کارایی را برای عملیات پایه اریه دهیم. به عبارت دیگر میخواهیم به سراغ تنظیمات برویم که m به مراتب کوچک‌تر از n باشد.