Hashing چیست؟ چرا از هشینگ استفاده می کنیم؟ قسمت ۱ (نسخه PDF)

Hashing چیست؟ چرا از هشینگ استفاده می کنیم؟ قسمت ۱ (نسخه چاپی)

Hashing چیست؟ بی شک واژه هش و هشـینگ را ممکن است روزانه بارها و بارها بشنوید و یا در صحبت های خود بکار برید. درباره ان نظر میدهیـد و نظر میشـنوید امـا خیلی کمتر از آن رغبت کرده ایـد خود را در دریـای مفهوم آن غرق کنیـد. در اینجا میخواهیم به عنوان توفیقی اجبـاری شـما را بـا مفهوم هش و هشـینگ بطور سـاده آشـنا کنیم. برای این موضوع فقـط کافیست کمی و فقط کمی از اطلاعات ریاضی خود را به ذهنتان فراخوانی کنید. با ما همراه باشید:

چرا هشینگ؟

اینترنت تشـکیل شـده از میلیون هـا انسان که روزانه چنـدین ترابایت محتوا تولیـد میکننـد. بر طبق سـرویس های جسـتجوی داده های اینترنت، مقدار محتوای تولید شده در اینترنت در هر شـش ماه، دو برابر میشوند. با وجود این رشد سـریع، جسـتجوی هرچیز در اینترنت غیر ممکن به نظر میرسد، مگر آن که ساختار و الگوریتم های جدیدی از داده را بر طبق مدل موجود، توسـعه دهیم تا بتوانیم داده ها را در اینترنت ذخیره سازی و به آن ها دسترسـی پیـدا کنیم. اما چرا ساختارهای سـنتی و مرسوم ذخیره سازی داده مانند اسـتفاده از آرایه ها و لیست های در هم لینک شده نمیتوانند جوابگوی نیاز امروزه باشند؟

فرض کنید داده زیادی را در اختیار داریم که آن ها را بر روی یک آرایه ذخیره کردیم. زمان تخمینی برای آن که پارامتری را در آرایه جستجو و پیدا کنیم میتواند O(log) و یا O(n) باشد. این زمان بسـتگی به این دارد که ارایه ما مرتب شده است یا خیر.اگر آرایه مرتب شده باشد، اسـتفاده از روش هایی چون جسـتجوی باینری میتوانـد در جسـتجوی آرایه مورد اسـتفاده قرار گیرد. درغیر این صورت آرایه باید بصورت خطی مورد جستجو قرار گیرد. تازه این در حالی است که ممکن است نتیجه حاصله از جستجوی حجم زیادی از داده مورد نظر ما نباشد.

بنابراین با توجه به موارد بالا، میخواهیم درباره تکنیک جدیدی به اسم هشینگ (Hashing) صحبت کنیم که این امکان را بما میدهد تا هر داده ورودی را زمـان ثابت O(۱) بروزرسـانی و بازیابی کنیم. زمان ثابت یا O(۱) به این مفهوم است که مقـدار زمـان لازم برای بازیابی داده بستگی به سایز داده (n) ندارد.

نقشه ساختار داده

تقسیم ساختار داده

در دنیای ریاضیات، یک نگاشت به ارتباط بین دو مجموعه اتلاق میشود. ما میتوانیم نگاشت M را به عنوان مجموعه ای از یک زوج در نظر بگیریم، جـایی که هر جفت بشـکلی از (Key, Value) است، در اینصورت برای کلیـد داده شـده، میتوانیم مقـداری را با اسـتفاده از نوعی عملکرد که کلیدها را به مقادیر متناظر میکند، پیدا کنیم. این کلید میتواند با استفاده از تابعی به نام تابع هش محاسبه شود. در ساده ترین حالت، میتوانیم تصور کنیم که آرایه ای بشـکل نگاشت خواهد بود اگر کلید بصورت یک شاخص قرار گیرد و مقدار بصورت ارزشـی برای آن شاخص. برای مثال آرایه A بما داده شده است.

اگر i کلیـد آن باشد، در اینصورت میتوانیم مقدار را با اسـتفاده از جسـتجوی A[i]، پیـدا کنیم. ایده اسـتفاده از جدول هش میتواند تعمیم یافته موارد بالا باشد که در ادامه آن را توضیح خواهیم داد.مفهوم جدول هش (Hash Table) ایده ای کلی برای یک آرایه است، زمانی که کلید الزاما یک مقدار صـحیح نیست. ما میتوانیم بجای کلید، یک نام و یا هر عنوانی در رابطه با شیء مورد نظر بگذاریم. هدف پیدا کردن تـابع هش برای محاسـبه یک شاخص است؛ در این صورت یک شـی میتوانـد در جای خاصـی از جـدول ذخیره شود مثلا جایی که بتواند براحتی پیدا شود.

- مثال : فرض کنید مجموعه از رشته های {"abc", "def", "ghi"} را در اختیار داریم که میخواهیم آن ها را در جـدول ذخیره کنیم. هدف ما در اینجا پیدا کردن و یا بروزرسانی این رشته ها از جدول و با استفاده از O(۱) است. نگرانی از بابت ترتیب آن ها یا هرگونه وجود نظم در آن ها به هیـچ وجه وجود نـدارد.فرض میکنیم "a"=۱،"b"=۲ و ... . همین ترتیب را برای تمـام حروف الفبـا در نظر میگیریم. سپس میتوانیم براحتی با این مکانیزم برای هرکدام از رشته ها بوسیله جمع مقدار عددی کاراکترها، یک عدد را محاسبه کنیم.

```
"abc"= 1+2+3=6,"def"=4+5+6=15,"ghi"=7+8+9=24
```

اگر ما فرض کنیم که جدولی با سایز ۵ برای ذخیره این رشـته ها داریم، میتوانیم مکان رشته را با استفاده از فرمول "sum mod ۵" محاسبه کنیم. بنابراین میتوانیم

- "abc" را در ۶mod ۵=۱
- "def" را در ۱۵mod ۵=۰
- و "ghi" را در ۲۴mod ۵=۴

یعنی مکان های ۱، ۰ و ۴ ذخیره کنیم.

در نهایت میخواهیم اگر به ما رشـته ای داده شـد، بتوانیم بسـرعت مکان آن را با اسـتفاده از یک تابع هش ساده که باقی مانـده تقسـیم مجموع کاراکترها بر سایز جدول است را محاسـبه کنیم. با اسـتفاده از این مقدار هش میتوانیم رشـته را جستجو کنیم. به نظر میرسد این روش راه خوبی برای ذخیره یک دیکشـنری باشـد. همچنین راه خوبی برای ذخیره زوج های (key,value) در جدول نیز هست.در قسمت بعد قصد داریم تا درباره مسائل و مشکلات موجود پیرامون هشینگ، پیدا کردن یک تابع هش خوب و اجرای ساده ای از یک جدول هش صحبت کنیم. در قسمت بعد با ما همراه باشید.سربلند و ITPro ای باشید.

پایان قسمت اول

نویسنده: احسان امجدی

منبع: انجمن تخصصی فناوری اطلاعات ایران

منبع: انجمن تخصصی فناوری اطلاعات ایران

مطلب اصلی