Bloom filtering :

این فیلتر به وسیله فردی به نام بلوم ارائه شد و یک Data structure احتمالاتی و تقریبی با پیچیدگی مکانی بسیار بهینه است. وقتی می خواهیم وجود عضوی را در مجموعه بررسی کنیم این Data structure بسیار می تواند مفید واقع شود.
فرض کنید می‌خواهیم وجود یک عنصری را در مجموعه بررسی کنیم، اگر Data structure به شما جواب داد که این عضو در مجموعه وجود دارد احتمال دارد که وجود نداشته باشد. اما اگر بگوید وجود ندارد، قطعاً درست هست. بر این اساس positive error داریم ولی negative error امکان پذیر نمی‌باشد.
این Data structure امکان درج عنصر را به ما می‌دهد ولی امکان حذف آن وجود ندارد. هرچه بیشتر عنصر به آن اضافه شود احتمال خطای مثبت بیشتر می‌شود. یعنی احتمال اینکه Data structure به شما بگوید عضوی در مجموعه وجود دارد اما در واقع وجود نداشته باشد بیشتر می‌شود.

پیچیدگی زمانی فیلتر بلوم O(k) (در عملیات درج عنصر، جستجوی عنصر) است که k به معنی تعداد تابع hash generator است.

الگوریتم

یک فیلتر بلوم خالی متشکل از یک آرایه ی m بیتی است که تمامی عناصر آن صفر است. همچنین به تعداد K تابع hash generator در اختیار هست که هرکدام از آنها یک عدد را به یکی از خانه‌های آرایه فیلتر بلوم نظیر می‌کند و خانه را از صفر به یک تغییر می‌دهند.

اضافه کردن عضو به فیلتر

برای این کار ابتدا عدد مورد نظر را به K تابع hash generator می‌دهیم. بر این اساس به تعداد K موقعیت در آرایه برای این عدد مشخص می‌شود. مقدار آرایه در موقعیت‌های مشخص شده را یک می‌کنیم. به این ترتیت عضو مورد نظر در فیلتر بلوم اضافه می شود.

جستجو در فیلتر

فرض کنید می‌خواهیم وجود یا عدم وجود عنصر x را در فیلتر بلوم بررسی کنیم. ابتدا این عنصر به K تابع hash generator داده و K موقعیت این عدد را در آرایه بلوم مشخص می‌کند. سپس مقادیر آرایه را در این K محل

بررسی می‌کنیم .اگر در تعداد یک یا بیشتری خانه مقدار صفر بود، فیلتر با قطعیت جواب می‌دهد که این عنصر در آرایه وجود ندارد، زیرا اگر بود باید تمامی این K محل یک می‌بود. اما اگر تمامی آنها یک بود، معلوم نیست که این یک شدن به خاطر وجود این عنصر هست، یا به خاطر درج‌های دیگر عناصر منجر به یک شدن این خانه شده‌است. بنابراین جواب می‌دهد که احتمالاً این عنصر در فیلتر حضور دارد. از اینجا معلوم می‌شود که هرچه بیشتر عضو در آرایه درج بشود، احتمال غلط بودن جواب وجود عنصر بیشتر می‌شود.

حذف عنصر از فیلتر

امکان حذف عنصر از فیلتر معمولی که از صفر و یک تشکیل شده است، نیست. یعنی چون عدم وجود عنصر باید با قطعیت گفته شود، امکان حذف نداریم. این امر از اینجا حاصل می‌شود که موقعی که مایلیم عنصری را حذف کنیم، یعنی باید تمام بیت‌های متناظر این عنصر را در آرایه صفر کنیم. به این ترتیب این عنصر حذف می‌شود. ولی ممکن است مکان‌های عددهای دیگر با این عنصر اشتراکاتی داشته باشد که به اشتباه آن خانه‌ها صفر شده‌است. بنابراین اگر بعد از حذف عنصر، قصد جستجوی عنصر دیگر را داشته باشیم، اگر فیلتر جواب دهد که این عنصر در آرایه وجود ندارد ممکن است جواب درستی نباشد. زیرا صفر بودن بعضی از موقعیت‌های این عنصر در آرایه می‌تواند به معنی عدم وجود این عنصر نباشد. در واقع در بلوم فیلتر ساده چنین امکانی وجود ندارد. اما اگر واقعا به چنین چیزی نیاز باشد، می توان از یک نسخه تغییر یافته بلوم فیلتر به نام Counting bloom filter استفاده کرد. در این فیلتر به جای مرتب سازی یک بیت تکی از مقدارها، یک مقدارinteger ذخیره می شود و بردار بیتی بردار integer خواهد بود. قابلیت حذف کردن باعث افزایش اندازه و هزینه خواهد شد. در این حالت به جای علامت گذاری یک مقدار بیت با ۱ هنگام ورود داده، مقدار integer را به مقدار ۱ افزایش می دهیم و اگر داده جدیدی وارد بلوم فیلتر شده بود، یک واحد دیگر به آن افزوده می شود، مثلا مقدار خانه اگر ۱ باشد با افزایش مجدد به ۲ تغییر داده می‌شود و در هنگام حذف کلمه یک واحد از خانه‌های خروجی توابع هش کم می‌شود. برای بررسی اینکه یک عنصر موجود هست یا نه کافیست که عنصر را هش کنیم و خانه هایی را که مورد بررسی قرار می گیرند مقدار بیشتر از صفر داشته باشند.

کاربرد bloom filtering

کاربرد فیلتر بلوم در کامپیوتر، کاهش بار سرور (کاهش پیچیدگی زمانی) و در عین حال استفاده اندک از حافظه است. به عنوان مثال اگر بخواهیم از درج رمزهای عبور نامناسب توسط کاربر جلوگیری کنیم، در این صورت باید کلمه به کلمه را در بانک اطلاعاتی جستجو کنیم. در حالت عادی پیچیدگی زمانی جستجوی خطی برابر است با O(n) . حال اگر داده ها از قبل مرتب شده باشند پیچیدگی زمانی برابر است با O(log(n))، اگر از قبل جدول هش ایجاد شده باشد پیچیدگی زمانی برابرO(1) است (این عمل به یک آرایه ثابت و رزرو شده نیازمند است که

حجم زیادی از حافظه را اشغال می نماید). در صورت استفاده از فیلتر بلوم، ابتدا کلیه رمزهای عبور نامناسب در فیلتر بلوم درج می گردند. برای اطمینان حاصل کردن از نامناسب نبودن رمزعبور، آن را در فیلتر بلوم بررسی می نماییم؛ اگر فیلتر بلوم جواب دهد که این کلمه در مجموعه وجود دارد، احتمال دارد که وجود نداشته باشد و در این صورت برای حصول اطمینان از نامناسب نبودن کلمه باید آن را در بانک اطلاعاتی بررسی نمود، اما اگر بگوید وجود ندارد، قطعاً درست هست و دیگر در بانک اطلاعاتی جستجو انجام نمی شود.

Quora و medium برای اینکه کاربران استوری ها و پست هایی که تابحال ندیده اند را مشاهده کنند و آنهایی را که قبلا دیده اند را نبینند، از bloom filtering استفاده می کند.

مرورگر Chrome برای شناسایی url های مخرب از bloom filtering استفاده می کند.

Google BigTable، Apache HBaseو Apache Cassandra و Postgresql از فیلترهای Bloom برای کاهش جستجوی دیسک برای سطرها یا ستونهای غیرقابل استفاده، بهره می برد. این باعث می شود تعداد درخواست ها به پایگاه داده بسیار کمتر شود.