Nr rejestracyjny: 2017/26/E/ST6/00191; Kierownik projektu: dr hab. Artur Jeż

Wybrane Zagadnienia Kompresji Gramatykowej streszczenie popularnonaukowe

Dane komputerowe zajmują coraz więcej miejsca — filmy mają coraz wyższą rozdzielczość, mp3 coraz wyższy bitrate, każdy ma coraz więcej znajomych na Facebooku, itd. A wszystko to musi być składowane na dyskach (nawet jeśli jest „w chmurze") przesyłane i przetwarzane. Dlatego większość takich danych jest *kompresowana*. Do różnych zastosowań używane są różne metody kompresji i w tym projekcie będziemy zajmować się jedną z nich — *kompresja gramatykową*.

Trochę trudno w paru zdaniach powiedzieć, skąd taka nazwa; łatwo za to wytłumaczyć, jak się kompresji gramatykowej używa: skompresowany plik podany jest w postaci zbioru reguł „`litera` $\to$ `ciąg liter`", np. $M \to miSP, S \to NN, N \to ssi, P \to pi$ oraz ciągu znaków, np. M. Aby odtworzyć oryginalną wiadomość, trzeba kolejno zastępować symbole tak, jak mówią reguły: najpierw wszystkie M przez $miSP$, potem S przez NN itd. W naszym przykładzie otrzymujemy słowo „mississipi" (nasz zapis nie wygląda na szczególnie krótszy, niż samo słowo, ale kompresje działa dobrze dopiero dla długich napisów i tak naprawdę reguły też się przedstawia inaczej). Tego typu kompresja jest podstawą standardu LZ78 i LZW, używanych np. w plikach graficznych GIF czy w plikach PDF.

Ale jak podać najmniejszą taką postać dla danego pliku? Okazuje się, że niestety tego nie wiadomo, co więcej, możemy *udowodnić*, że nie jesteśmy w stanie szybko obliczyć najlepszego sposobu skompresowania. Z drugiej strony, jest wiele metod, które próbują kompresować może nie w najlepszy, ale możliwie dobry sposób. W tym projekcie będziemy zajmować się, między innymi, analizą najpopularniejszych takich metod — pokazać, jak dobrze i jak źle potrafią one skompresować pliki. Chcemy też zaproponować nowe standardy kompresji gramatykowej, pozbawione niektórych wad poprzednich wariantów.

Można zapytać: skoro wiadomo, że nie potrafimy skompresować tą metodą w najlepszy możliwy sposób, to dlaczego w ogóle jej używamy? Po pierwsze, to w praktyce nie jest tak źle i działa ona całkiem dobrze. Po drugie kompresja gramatykowa ma ważną zaletę: na pliki skompresowanych tą metodą można dość łatwo przekształcać, bez potrzeby dekompresji. Pomyślcie o skompresowanym filmie: można go obejrzeć czy przewijać i nie wymaga to rozpakowania go w całości; w plikach skompresowanych kompresją gramatykową można wyszukiwać, edytować, wycinać ... prawie tak szybko, jakby tej kompresji wcale nie było i to niezależnie od tego, o jakich danych mówimy — muzyce, plikach tekstowych czy prezentacjach. Takie podejście znalazło zastosowanie w nauce — są całe klasy problemów, dla których najlepszym sposobem rozwiązania, jaki znamy, to skompresowanie zadania i liczenie rozwiązania na tej mniejszej, skompresowanej postaci. Tego typu obliczenia stosuje się np. równań w grupach. Tym też będziemy się zajmować — chcemy podać kolejne rozwiązania używające tego podejścia.

Jak już powiedzieliśmy, znamy bardzo szybkie algorytmy działające na skompresowanych danych. Ale nie wiadomo, może nie są one najlepsze i należy je ulepszyć? Ale może tak się po prostu nie da? Tym też będziemy zajmować się: *granice dolne* to ścisły, matematyczne argumenty na to, że czegoś nie da się zrobić lepiej lub szybciej. Będziemy pokazywać takie granice dla różnych problemów dla skompresowanych danych, zwłaszcza dla tych, które są stosowane w innych działach nauki.

Wszystko, co na razie powiedzieliśmy, dotyczy kompresji gramatykowej danych traktowanych jako strumień znaków. W pewnym sensie jest to najbardziej ogólne, bo koniec końców wszystko, co komputer przetwarza, to ciąg znaków, a nawet ciąg zer i jedynek (bitów). Niestety, wiele danych ma dodatkową strukturę, która jest skutecznie ukrywana przy okazji kodowania w postaci bitów. Na szczęście, dla innych formatów też istnieją odpowiednie kompresje gramatykowe, ale tu sytuacja staje się dużo bardziej skomplikowana: możliwości jest dość dużo i nie bardzo wiadomo, która jest lepsza. W tym projekcie zajmiemy się porównaniem kilku modeli kompresji gramatykowej dla najprostszych struktur, które nie są ciągami — tzw. struktur drzewowych. Porównamy, jak dobrze kompresują i jakie algorytmy można wykonać na ich skompresowanej postaci.