Sidharth Ghoshal
August 4, 2016
In principle there should be a mistake somewhere in here, for this problem cannot be that easy. Still working on finding that error, in the meanwhile here it is for documentation. Please try to find the error that I cannot, and email sid[dot]Ghoshal[at]yahoo.com if you do.

This algorithm was actually invented last year but was not given a very serious analysis until now:

On a New Breed of 0-1 Integer Programming Algorithms, and a mysterious proof that $P = NP$

Conjecture: Given two $n-1$ dimensional polyhedral P_1, P_2 in $\mathbb{R}^n$ whereas P_1, P_2 are embedded in parallel hyperplanes then the convex hull of P_1, P_2 is the cobordism of lowest number of $(n-1)$ facets of all piecewise linear cobordisms between the two. Additionally, this convex hull can be computed in polynomial time with respect to the number of (n-2) facets in P_1, P_2

A simple statement of the problem:

Given $Ax \leq b, 0 \leq x_i \leq 1$, determine if there are any feasible integer points.

Solution: Algorithm Y

Current System $Ax \leq b, 0 \leq x_i \leq 1$
for each x_i DO:
branch into two systems based on $x_i = 1, x_i = 0$ (Called System 1 and System 0)
Remove redundant constraints
If a system is found to be inconsistent, skip formation of Hull, and continue algorithm on next variable.
Compute the convex hull of these by:
Taking each inequality $A_n x \leq b_n$ in System j, maximize $A_n x$ in System $1-j$, to yield a point p_n
Observe that a unique hyperplane can be formed that intersects
$(A_n x = b \wedge x_i = j), p_n$
Of the form $H_n x = \tau_n$
Then $H_n x \leq \tau_n$
Is a constraint in the convex hull
Replace $Ax \leq b \; with \; Hx \leq \tau$
Remove redundant constraints
End Solution

Claim: branching for a SINGLE variable takes polynomial time in the number of variables and inequalities (trivial)

Claim: maximizing edges takes time polynomial in the number of variables and edges, in the worst case say we have n variables and m inequalities. Then at the branch step each half of the branch has m inequalities, so that means $2m$ maximizations occurred which using a polynomial time LP algorithm would take $O(m^{4.5}n^{3.5})$ time.

Now if the conjecture is true, then the resultant hull after removing redundant constraints will have less than or equal to m inequalities (and variables haven't changed).

As a result we can conclude that this runs in time $O((mn)^{4.5})$ (since it runs once for each variable and there are n)

Claim: if a 0-1 IP has no integer solutions, then algorithm Y must necessarily conclude there are no solutions to the system.

Proof:

Definition:

for a variable x_j, let two integer points u_1, u_2 be "adjacent" if they differ only on their j^{th} coordinate.

One can consider the edge between u_1, u_2.

Lemma:

if at least one of two j-adjacent points u_1, u_2 is not feasible then after x_j is processed, no point on the edge (as an open interval) will be present in the system after Algorithm Y iterates through x_j

Proof:

the branching of this edge by x_j is u_1 and u_2, so in order for it to enter the convex hull BOTH must be feasible points

Generalization:

for a variable x_j two convex sets of points $U_1, U_2$are considered "adjacent" if there exists a bijection $\phi: U_1 \to U_2$ such that each pair $\tau \in U_1, \phi(\tau) \in U_2$ differ only in their j^{th} coordinate.

Lemma:

If at least one of two j-adjacent sets U_1, U_2 is entirely not feasible, then the convex hull of U_1, U_2 excluding its boundary will not be feasible in the system after Algorithm Y iterates through x_j

Proof:

Branching the convex hull of both sets yields the convex hull of U_1 and convex hull of U_2. Of course these were convex to begin with, so they are their own convex hull, and it follows if one of them was never feasible then only part of the hull that may be feasible must be in the boundary (ie contained in U_1, U_2).

So having established this suppose we have a 0-1 IP of n variables and m equations that does not contain any integer solutions called Q.

Lemma:

Then at the start of stage i of Algorithm Y on Q there will be 2^{n-i} convex hulls of 2^i integer points, that are definitely infeasible and adjacent by x_i.

This can be proven by induction,

Base Case: stage 0, 2^n individual integer points are infeasible

Inductive Step: Suppose this is true for stage i-1, We note exactly half of all these hulls lie on $x_i = 0$ and the other half lie on $x_i = 1$, and there is a natural bijection ϕ between the two branches. From here since both hulls are infeasible we use our earlier Lemma to conclude that after the algorithm passes, the hull of each pair of hulls must be infeasible, which of course has twice as many points and so it follows that there are now 2^{n-i} convex hulls of 2^i integer points.

Of course that also implies that at the final round we will have 1 hull of all the points that is infeasible.

So we have a polynomial time algorithm for checking feasibility, there exists then a trivial approach using binary search on the objective function to conclude that general 0-1 Integer Programming can now be done in polynomial time, implying P=NP, and the world collapses.

Of course that is ridiculous, what could happen in theory is that as the hulls are being formed the number of inequalities is spiking up. The cobordism conjecture essentially decides whether this is polynomial time or not. In the meanwhile we can test this experimentally and see what happens.