Polynomial Hierarchy collapses!

Theorem: If $\Pi_2 = \Sigma_2$ then $\Sigma_3 \subset \Sigma_2$.

- notations:
  - "$\Sigma_2 - formula$" :=formula of type $\exists x \forall y \phi(x, y)$
  - "$\Pi_2 - formula$" :=formula of type $\forall x \exists y \phi(x, y)$
  - "$\Sigma_3 - formula$" :=formula of type $\exists x \forall y \exists z \phi(x, y, z)$

given $L \in \Sigma_3$, we will show the existence of a poly-time machine,

that for each $x \in L$ produces a formula in the form of $\exists x \forall y \phi(x, y)$ such that this formula is valid iff $x \in L$.

Proof:

- for each $x \in L$, we can compute a formula $\exists x \forall y \exists z \phi^{'}(x, y, z)$ such that it is valid iff $x \in L$.

  this can be computed using turing machine denoted as M.

- then, denote the formula - $G(y, z) = \forall y \exists z \phi^{'}(x, y, z)$ where x is free.

  now, this is **not** a "$\Sigma_2 - formula$", because we have a free variable x here.

- so, we will create new turing machine M_0, that given $G(y, z)$ formula, and *assignment* for x,

  computes new $logically - equivalent$ formula, (where x is with that assignment) , with **no** x as a free variable in it.

  it will just do some replacments.

  for example, if $x = 1$, and $G(y, z) = (y \vee z) \wedge (x \vee y)$, then $M_0(G(y, z), x) = (y \vee z)$

  this will be turing machine M_0.

- then, for each assignment for x we can compute M_0 on $G(y, z)$ and x, and get a new "$\Pi_2 - formula$".

  call it $M_0(G(y, z), x)$.

- then we can evaluate "$\Sigma_2 - formula$" from it, in polynomial time, using turing machine M_1.

  that is because $\Pi_2 = \Sigma_2$, thus $M_0(G(y, z), x) \in \Sigma_2$.

  BUT, it is not enough, we need to have this turing machine M_1 **one** for all formulas of type "$\Pi_2 - formula$".

  - that is because, the formula $M_0(G(y, z), x)$ **varies** and depends on x. so we want to use the same turing-machine for all x.

but, we have that, because if $\Pi_2 = \Sigma_2$, then we have a **one** reduction poly-time computable, from **any** "$\Pi_2 - formula$" to an appropriate "$\Sigma_2 - formula$".

This reduction function will be called M_1.

- thus, $M_1(M_0(G(y,z),x))$ is in the type of $\exists x' \forall y' \phi(x',y')$.

  we will take only the body of this formula:$\phi(x',y')$,

  this can be surely also computed at a poly-time.

- note also that $|x'|, |y'| \leq |\{0,1\}^{p(|x|)}|$, where x is our original input, and p is some polynom.

  because all our computations are polynomial-time of the input.

  we can do that and replace all occurences of x', y' with x'', y'' for example, it does not matter.

- so, we will call the finally evaluted formula $\phi(x',y')$.

  this formula depends on x thus we can not write that the formula will be $\exists x. M_1(M_0(G(y,z),x))$ and so it will be $\exists x. \exists x' \forall y' \phi(x',y')$ and thats it,

  because it is not true! the formula ϕ **varies** with each assignment on x.

- **but, what we can do is that:**

  we will say that, $M_1 \circ M_0$ will compute only the body, and we will write in the begining of the formula: $\exists x \in \{0,1\}^{p(|x|)} \exists x' \in \{0,1\}^{p(|x|)} \forall y' \in \{0,1\}^{p(|x|)}$.

  then, the formula it-self, will be the $Cook - Levin$ computation for x as an input, applying M_0 on x and $G(y,z)$, then applying on the output M_1,

  creating $\phi(x',y')$, then M_2 that computes the logical value of $\phi(x',y')$, where x', y' as its free variables, inputs, and checking actually if M_2 output was $true$. so, we actually write the formula: $M_2(M_1(M_0(G(y,z),x)),x',y') = true$

- then the final formula is:

  $\exists x \in \{0,1\}^{p(|x|)} \exists x' \in \{0,1\}^{p(|x|)} \forall y' \in \{0,1\}^{p(|x|)} M_2(M_1(M_0(G(y,z),x)),x',y') = true$

  note that this formula is not depend on anything but the original formula which is: $\exists x \forall y \exists z \phi^{'}(x,y,z)$.

- the final formula is $\Sigma_2 - formula$.