Universidad de los Andes > FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

Entrega 0, Proyecto 2
Metodos Computacionales en OOCC, IOC 4201

Profesor:
Patricio Moreno

Ayudante:
Maximiliano Biasi

Alumno:
Lukas Wolff Casanova

22 de octubre de 2024

Índice

Índice de figuras

Entrega 0

Actualmente, el análisis de estructuras se realiza computacionalmente. Esto no solo ayuda a realizar los cálculos más rápido, sino que también permite llevar a cabo múltiples diseños en un corto período de tiempo. Esta sección tiene como objetivo aprender sobre la librería OpenSees y el análisis que hay detrás de un reticulado. Para esto, se realizó un análisis manual y se comparó con el análisis realizado en OpenSees.

El reticulado de referencia es el siguiente:

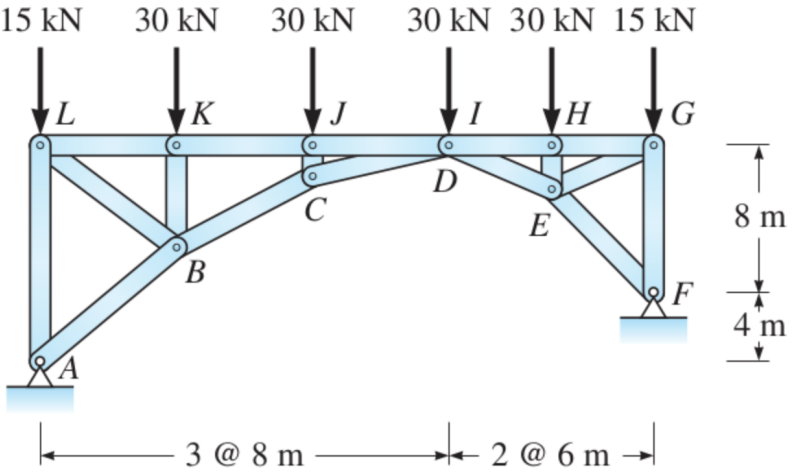

Figura 1: Reticulado

1. Material Utilizado

Para el cálculo de deformaciones, se consideró un módulo de elasticidad de 200 GPa y una fluencia de 210 MPa. Además, el mínimo factor de seguridad aceptado fue determinado en 1.3.

2. Comandos Para Correr Arquitectura X86 en Arquitectura ARM

Dado que trabajo en Mac, la arquitectura utilizada es ARM64, mientras que el requerimiento para las librerías es X86. Por lo tanto, se debe correr una instancia de X86 en la arquitectura ARM. Para activar el entorno, se utilizan los siguientes comandos:

Primero, se llama la instancia en una arquitectura X86:

```
arch -x86_64 python3 -m venv x86_env
```

Luego, se activa:

```
source x86_env/bin/activate
```

Para instalar las librerías necesarias:

```
arch -x86_64 pip install <librerías>
```

Finalmente, se debe correr el código desde la instancia:

```
arch -x86_64 python3 <ruta>.py
```

3. Cálculos Manuales

A continuacion se detallaran los resultados obtenidos a lo largo del proceso de calculo manual, donde el codigo para calcular el reticulado esta en siguiente enlace.

Para resolver el reticulado, se realiza como si fuera manuelamente, haciendo equilibrio nodo por nodo. Personalmente, utiliza python ya que ante cualquier error o cambio a realizar, es mucho mas rapido y no tengo que calcular todo nuevamente.

3.1. Calculo de Reacciones

Al calcular las reacciones en los apoyos, se obtiene lo siguiente:

Nodo	Reacción X [KN]	Reacción Y [KN]
A	64.286	72.857
F	-64.286	77.143

Cuadro 1: Reacciones en los Apoyos

3.2. Cálculo de alturas

Para calcular la altura de los puntos desconocidos, se determinó el momento en tal nodo igual a 0, obteniendo así:

Nodo	Altura [m]
B	7.733
C	11.733
E	9.400

Cuadro 2: Alturas Nodos Desconocidos

3.3. Solución Reticulado

Para calcular la sección de las barras, se utilizó el siguiente código, donde se itero cada seccion hasta que cumpliera con un FS minimo de 1.3.

Barra	Esfuerzo Interno	D_{int}, D_{ext} [mm]	Tensiones Internas	Fuerza Crítica Pandeo
AB	89.411	(117.000, 135.500)	24.371	117.169
AL	15.000	(85.000, 95.000)	10.610	19.682
LK	0.000	(10.000, 20.000)	0.000	0.227
LB	0.000	(10.000, 20.000)	0.000	0.177
BC	71.874	(98.500, 114.500)	26.852	94.163
BK	30.000	(52.000, 62.000)	33.506	39.731
KJ	0.000	(10.000, 20.000)	0.000	0.227
JC	30.000	(10.000, 20.000)	127.324	204.387
JI	0.000	(10.000, 20.000)	0.000	0.227
CI	64.321	(90.000, 105.000)	27.999	84.599
IH	0.000	(10.000, 20.000)	0.000	0.404
IE	70.062	(77.500, 93.500)	32.604	91.438
HE	30.000	(36.000, 46.000)	46.582	40.103
HG	0.000	(10.000, 20.000)	0.000	0.404
GE	0.000	(10.000, 20.000)	0.000	0.340
GF	15.000	(63.500, 73.500)	13.941	19.569
EF	86.488	(92.500, 110.500)	30.137	112.836

Cuadro 3: Información Barras

Nota: Todas las barras se encuentran en esfuerzo de compresión, ademas se asume la fluencia igual en traccion y comprecion.

Barra	FS Fluencia	FS Pandeo
AB	8.617	1.310
AL	19.792	1.312
BC	7.821	1.310
BK	6.267	1.324
JC	1.649	6.813
CI	7.500	1.315
IE	6.441	1.305
HE	4.508	1.337
GF	15.064	1.305
EF	6.968	1.305

Cuadro 4: Factor de Seguridad Fluencia y Pandeo

Nota: Solo se consideran las barras que tienen algún esfuerzo interno significativo para el cálculo del FS.

3.4. Cálculo de Deformación

Para el cálculo de la deformación en el nodo D/I, se impone una fuerza virtual igual a 1. De esta manera, es posible calcular el reticulado. Con todos los esfuerzos conocidos, se aplica la siguiente fórmula:

$$\delta = \sum \frac{P_r \cdot P_v \cdot L}{E \cdot A} \tag{1}$$

Se obtuvieron los siguientes desplazamientos, donde la sección considerada es (117, 135.5):

$$\delta_x = -8{,}47300012321403mm \quad \delta_y = -10{,}1170948633426mm \tag{2}$$

El código para calcular el desplazamiento horizontal está en el siguiente enlace, mientras que el código para el desplazamiento vertical está en el siguiente enlace.

4. OpenSees

Para realizar el modelo en OpenSees, se agregó la barra KC, de modo que el sistema no fuera hiperestático. Aun así, esto no afectó el cálculo, ya que dicha barra no tiene esfuerzo interno considerable.

4.1. Solución Reticulado

El código utilizado para hacer el modelo en OpenSees se encuentra en el siguiente enlace, y los esfuerzos obtenidos son los siguientes:

Cuadro 5: Esfuerzos internos en las barras (fuerzas axiales)

Barra	**Fuerza Axial (KN)**
AL	15.03
AB	89.39
LB	0.06
LK	0.05
KB	30.03
BC	71.82
KJ	0.79
KC	0.74
JC	30.00
CD	63.53
JI	0.79
IH	0.00
IE	70.06
HE	30.00
EF	0.00
HG	0.00
EF	86.49
GF	15.00

Nota: Se observa claramente una relación directa con los resultados manuales, donde la pequeña variación nace de la adición de la barra **KC**.

4.2. Deformaciones

Para el cálculo de las deformaciones, se asume una sección igual para todas las barras. Según los datos calculados manualmente, la sección necesaria es (117, 135.5), y los resultados obtenidos son los siguientes:

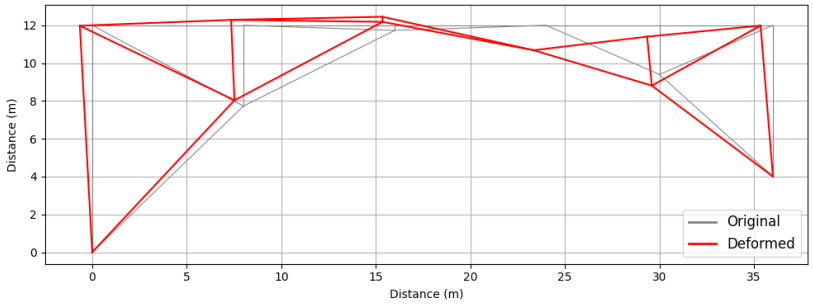

Figura 2: Deformaciones en los nodos

El diagrama tridimensional solicitado es el siguiente:

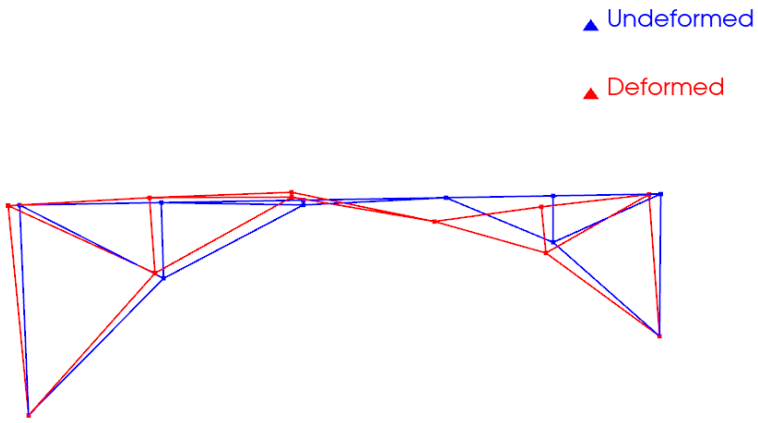

Figura 3: Gráfico 3D

La deformación observada en el nodo D/I es:

$$\delta_x = -0{,}0065527383052691m \quad \delta_y = -0{,}013182996668814742m \tag{3}$$

De esta manera, los porcentajes de error respecto al cálculo manual son:

$$\delta_x = 29{,}31\,\% \quad \delta_y = 30{,}23\,\% \tag{4}$$

5. Conclusión

En conclusión, los objetivos de la presente entrega fueron logrados correctamente. Se logró generar un análisis del reticulado de manera manual, donde se determinaron los distintos factores de seguridad, además de las secciones mínimas que debe tener cada barra. Tomando como referencia la barra de mayor área e inercia, se determinó el desplazamiento en el nodo solicitado mediante fuerzas virtuales.

Posteriormente, se realizó el mismo análisis en OpenSees, donde se observó que los resultados no variaron significativamente. Se determinó el desplazamiento en el nodo D y se generaron los gráficos solicitados.

Finalmente, se cuantificó el error entre ambos métodos, comparando la variación del desplazamiento calculado. Considerando las pequeñas variaciones en el cálculo, el error está dentro de un rango esperado.

6. Anexos

El razonamiento utilizado para el código OpenSees es el siguiente:

```
    # Se define un modelo 2D con 2 grados de libertad
    ops.wipe()
    ops.model('basic', '-ndm', 2, '-ndf', 2)
```

```
    # Definir el material uniaxial elástico, con módulo de elasticidad E
    matTag = 1  # Material tag
    ops.uniaxialMaterial('Elastic', 1, E)
```

```
    # Se definen los nodos
    nodes = {
    1: (0, 0),
    2: (0, 12),
    3: (8, 7.73),
    4: (8, 12),
    5: (16, 11.73),
    6: (16, 12),
    7: (24, 12),
    8: (30, 9.4),
    9: (30, 12),
    10: (36, 12),
    11: (36, 4)
    }
    # Generar los nodos
    for node_id, coords in nodes.items():
        ops.node(node_id, *coords)
```

```
    # Definir los elementos truss
    elements = [
        (1, 2), #1
        (1, 3), #2BC
        (2, 3), #3
        (2, 4), #4
        (3, 4), #5
        (3, 5), #6
        (4, 6), #7
        (4, 5), #8
        (5, 6), #9
        (5, 7), #10
        (6, 7), #11
        (7, 9), #12
        (7, 8), #13
        (8, 9), #14
        (8, 10), #15
        (9, 10), #16
        (8, 11), #17
        (10, 11) #18
    ]

    for i, (ni, nj) in enumerate(elements, start=1):
        ops.element('Truss', i, ni, nj, A1, matTag)
```

```
    # Se definen las restricciones
    ops.fix(1, 1, 1)
    ops.fix(11, 1, 1)
    # El nodo 1 y el nodo 11 son fijos en x e y
```

```
    # Definir cargas en los nodos
    ops.timeSeries('Constant', 1)
    ops.pattern('Plain', 1, 1)
    ops.load(2, 0.0, -15000.0)
    ops.load(4, 0.0, -30000.0)
    ops.load(6, 0.0, -30000.0)
    ops.load(7, 0.0, -30000.0)
    ops.load(9, 0.0, -30000.0)
    ops.load(10, 0.0, -15000.0)
```

```
    # Se resuelve el modelo
    ops.system('BandSPD')
    ops.numberer('RCM')
    ops.constraints('Plain')
    ops.integrator('LoadControl', 1.0)
    ops.algorithm('Linear')
    ops.analysis('Static')

    # Ejecutar análisis
    if ops.analyze(1) != 0:
        print("Error en el análisis estático")
    else:
        print("Análisis estático exitoso")
```