

On the Fine-Grained Complexity of Parity Problems

Amir Abboud

Shon Feller

Oren Weimann

Conditional Lower Bounds

Start with a popular conjecture:

Problem X cannot be solved in $O(n^{\delta-\varepsilon})$ time.

Reduce problem X to your problem in $O(n^{\delta-\varepsilon})$ time.

Your problem cannot be solved in $O(n^{\delta-\varepsilon})$ time.

All Pairs Shortest Paths (APSP)

Input: An edge-weighted graph G .

Output: Return all distances $d(u, v)$ in G .


No $O(n^{3-\varepsilon})$ algorithm for any $\varepsilon > 0$

Negative Weight Triangle (NWT)

Input: An edge-weighted graph G .

Output: Is there a triangle in G with total negative weight?

NWT can be solved in $O(n^{3-\varepsilon})$ iff APSP can be solved in $O(n^{3-\varepsilon'})$ [Vassilevska-Williams, Williams '10]



Subcubic
equivalent

All Pairs Shortest Paths (APSP)

Input: An edge-weighted graph G .

Output: Return all distances $d(u, v)$ in G .

No $O(n^{3-\varepsilon})$ algorithm for any $\varepsilon > 0$

Subcubic
equivalent

Negative Weight Triangle (NWT)

Input: An edge-weighted graph G .

Output: Is there a triangle in G with total negative weight?

NWT can be solved in $O(n^{3-\varepsilon})$ iff APSP can be solved in $O(n^{3-\varepsilon'})$ [Vassilevska-Williams, Williams '10]

3-SUM

Input: A list $A[1..n]$ of n integers.

Output: Are there i, j, k s.t $A[i] + A[j] + A[k] = 0$

No $O(n^{2-\varepsilon})$ algorithm for any $\varepsilon > 0$

Min-Plus Convolution

Input: Two lists $A[1..n], B[1..n]$ of n integers each.

Output: Return vector C where $C[k] = \min_{i+j=k} A[i] + B[j]$

No $O(n^{2-\varepsilon})$ algorithm for any $\varepsilon > 0$. If APSP can be solved in $O(n^{3-\varepsilon})$ or 3-SUM can be solved in $O(n^{2-\varepsilon})$ then Min-Plus Convolution can be solved in $O(n^{2-\varepsilon'})$. [Bremner et al. '12], [Cygan et al. '17]

Known results in fine-grained complexity

- Problems subcubic-equivalent to APSP:
NWT, Min-Plus Product, Radius, Median, Maximum 2D Subarray, Metricity Testing, Wiener Index, Second Shortest Path, ...
- Problems subquadratic-equivalent to Min-Plus Convolution:
0/1 Knapsack, Maximum Consecutive Subsums, Approximation of Subsets Sums, ...
- 3-SUM based conditional lower bounds for problems in computational geometry, dynamic problems, ...

Our results

- We consider the *parity versions* of classical problems studied in fine-grained complexity.
- ***Parity computation***: Is the solution even or odd?
Diameter parity: Return whether the diameter is even or odd.
- ***Parity counting***: Is the number of solutions even or odd?
NWT parity: Return whether the number of negative triangles is even or odd.
- We show that for many classical problems finding the exact solution is as hard as finding its parity.

Our results (in bold)

APSP-related results:

Problem	Complexity
APSP	Parity is Subcubic Equivalent to APSP
Median	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Radius	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Sum of Eccentricities	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Integer Betweenness Centrality	Subcubic Equivalent to APSP, Approximation is Subcubic Equivalent to Diameter under randomized reductions, Parity is Subcubic Equivalent to APSP
Second Shortest Path	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Weiner index	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP

Problem	Complexity
Maximum Subarray	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Min-plus matrix multiplication	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Replacement Paths	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Negative Weight Triangle	Subcubic Equivalent to APSP, Approximation counting is equivalent to APSP under randomized reductions Vertex parity is Subcubic Equivalent to APSP, Randomized reductions from APSP and 3SUM to parity counting
Zero Weight Triangle	Reductions from APSP and 3SUM, Randomized reduction to parity counting and vertex parity

Our results (in bold)

Other results:

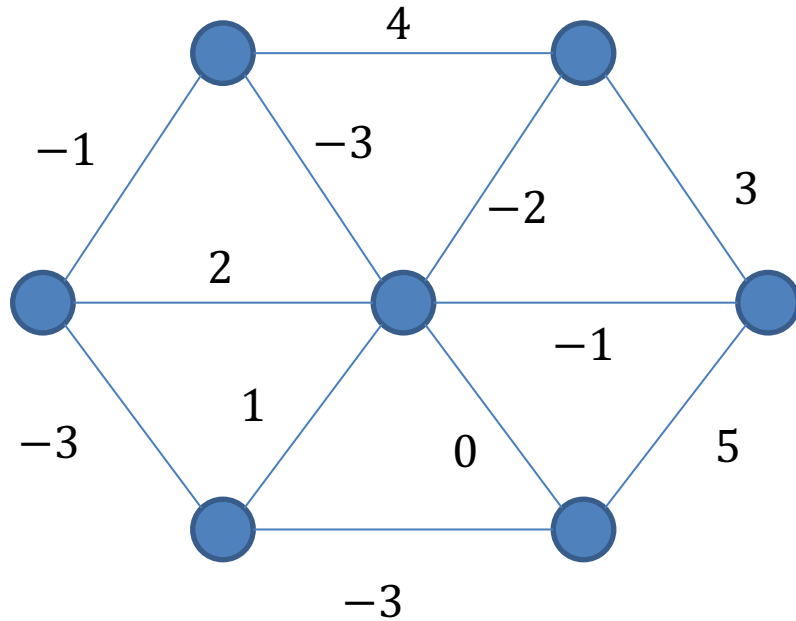
Problem	Complexity
Diameter	Parity is Subcubic Equivalent to Diameter
Min-plus convolution	Reductions to APSP and 3SUM, Parity is subquadratic equivalent to Min-plus convolution
0/1-Knapsack	Subquadratic equivalent to Min-plus convolution under randomized reductions Parity is subquadratic equivalent to Min-plus convolution
Reach Centrality	Subcubic Equivalent to Diameter, Parity is Subcubic Equivalent to Diameter

Problem	Complexity
Maximum Row Sum	Reduction from Co-Negative Triangle to parity
Tree sparsity	Subquadratic equivalent to Min-plus convolution under randomized reductions Parity is subquadratic equivalent to Min-plus convolution
Maximum Consecutive Subsums	Subcubic Equivalent to Diameter, Parity is Subcubic Equivalent to Diameter
Co-Negative Triangle	Reduction to Diameter, Reduction to Maximum Row Sum Parity

Negative Triangle Vertex Parity (NTVP)

Input: An edge-weighted graph.

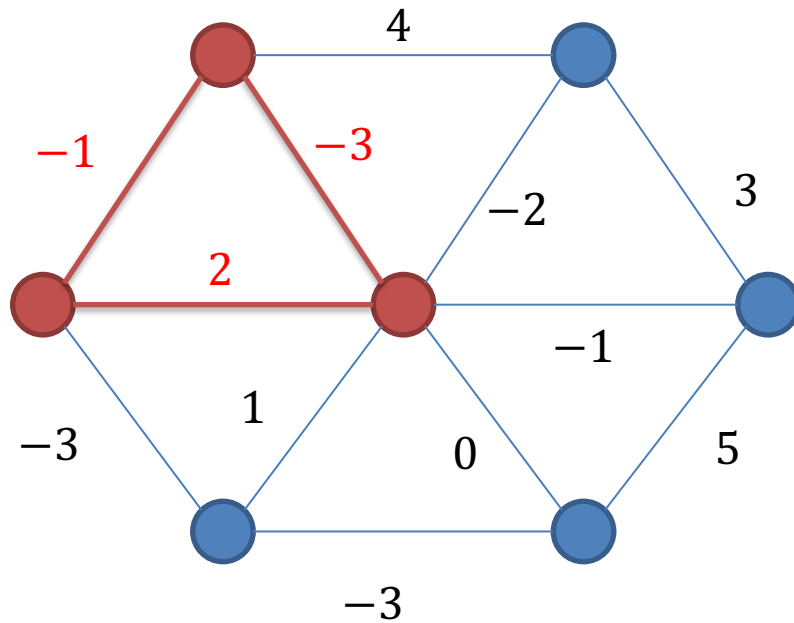
Output: The parity of the number of vertices belonging to a negative triangle.



Negative Triangle Vertex Parity (NTVP)

Input: An edge-weighted graph.

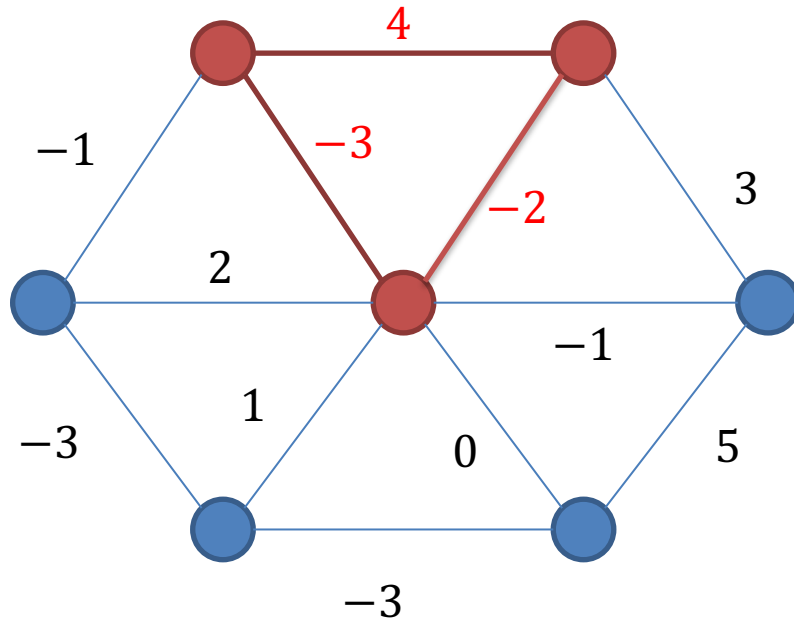
Output: The parity of the number of vertices belonging to a negative triangle.



Negative Triangle Vertex Parity (NTVP)

Input: An edge-weighted graph.

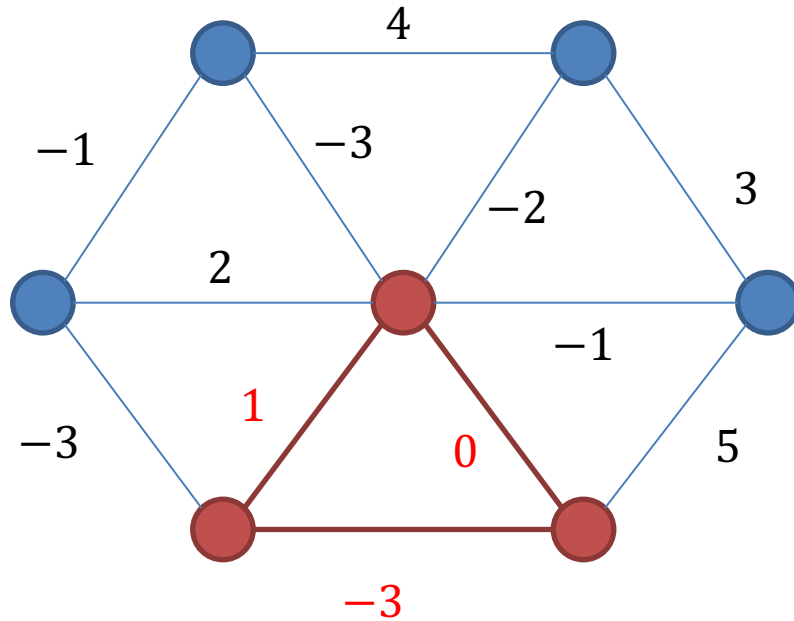
Output: The parity of the number of vertices belonging to a negative triangle.



Negative Triangle Vertex Parity (NTVP)

Input: An edge-weighted graph.

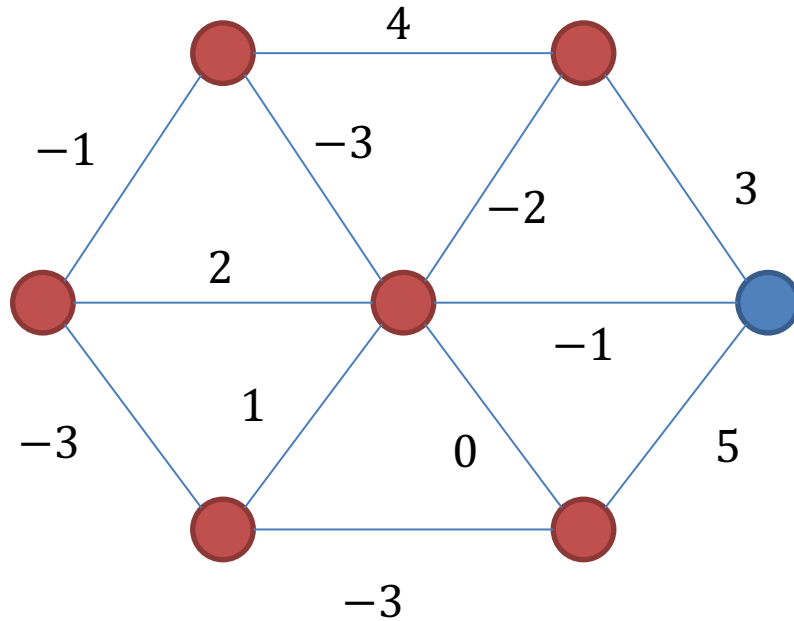
Output: The parity of the number of vertices belonging to a negative triangle.



Negative Triangle Vertex Parity (NTVP)

Input: An edge-weighted graph.

Output: The parity of the number of vertices belonging to a negative triangle.

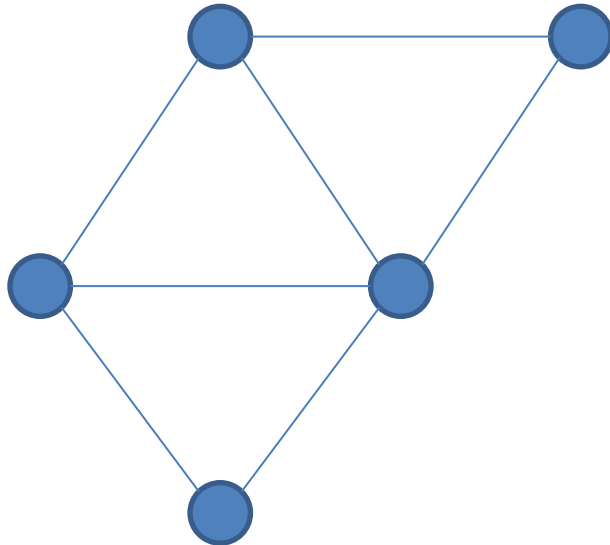


Parity = 0 (even)

Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

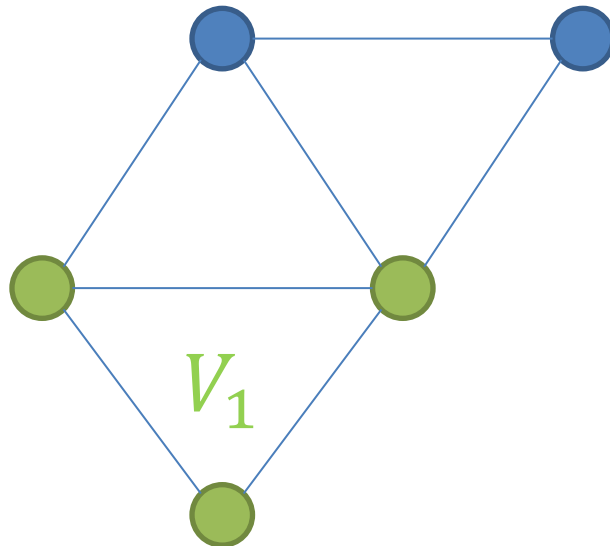
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

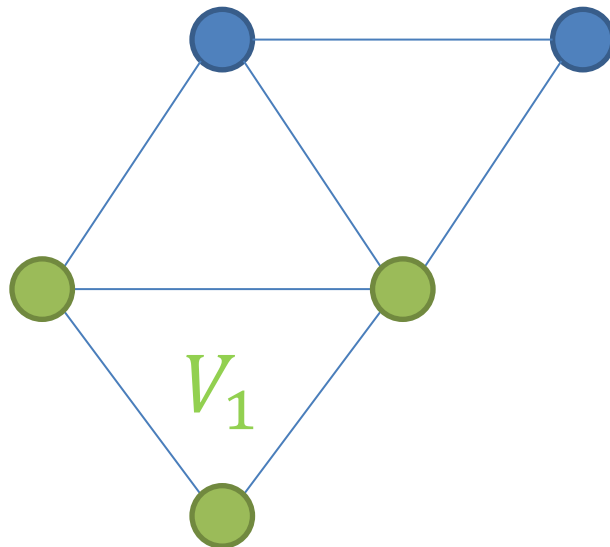
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

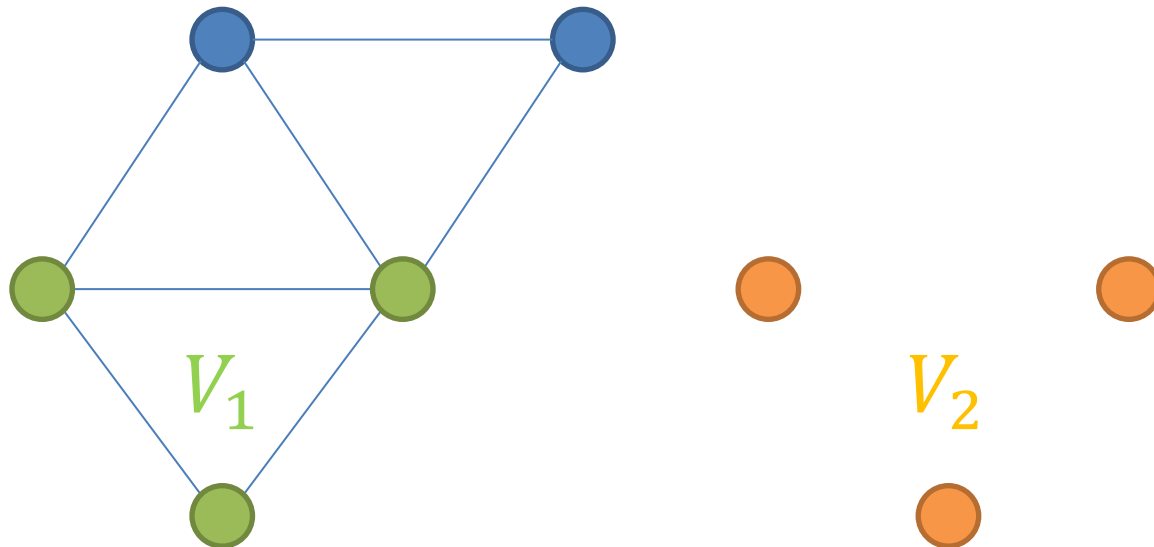
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

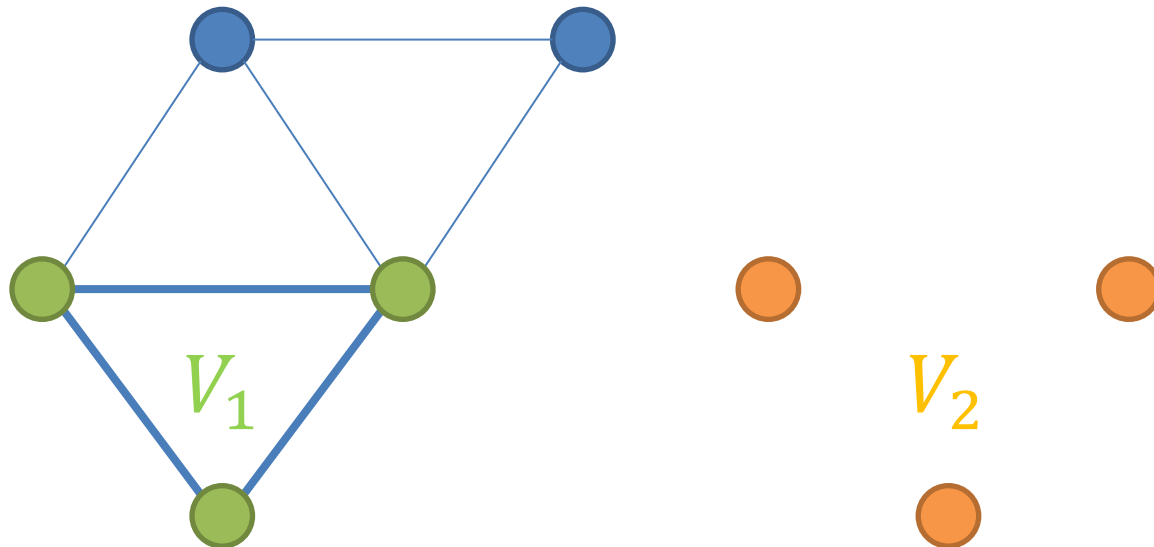
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

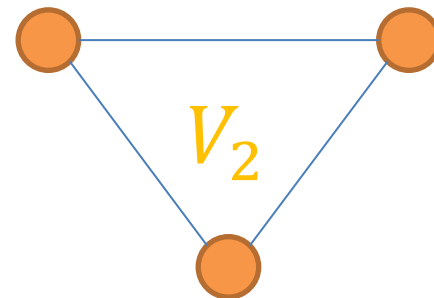
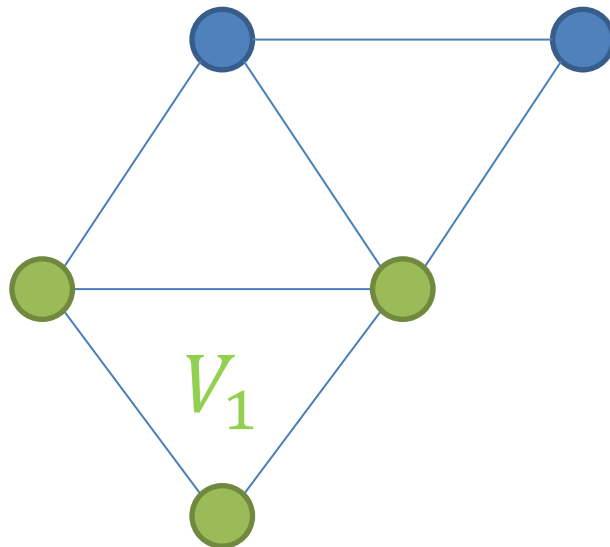
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$
- For every $(u_1, v_1) \in V_1 \times V_1$ add $(u_2, v_2) \in V_2 \times V_2$



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

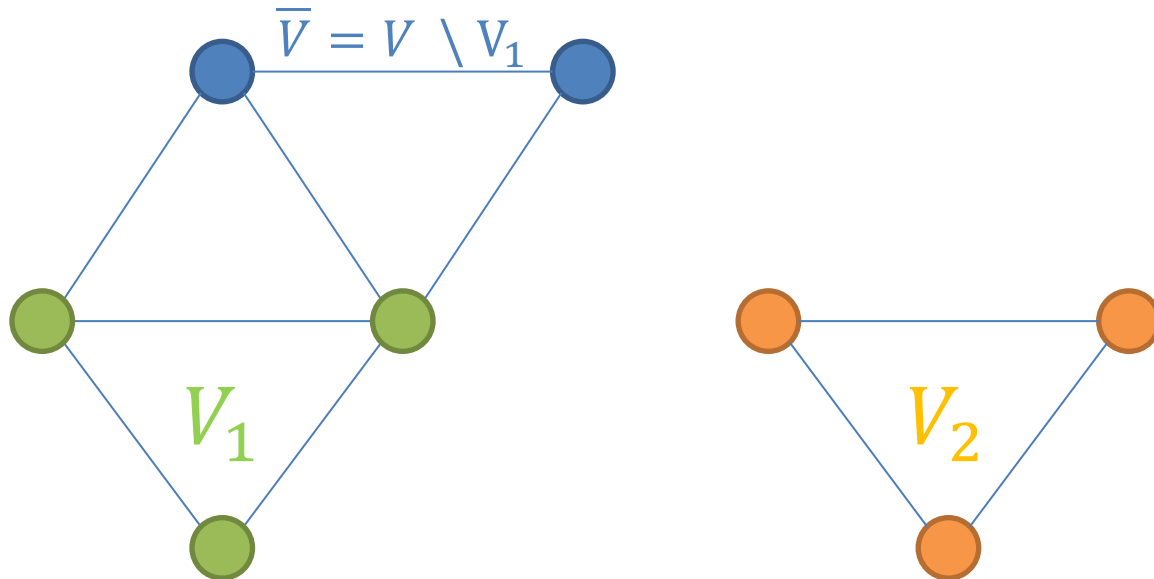
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$
- For every $(u_1, v_1) \in V_1 \times V_1$ add $(u_2, v_2) \in V_2 \times V_2$



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

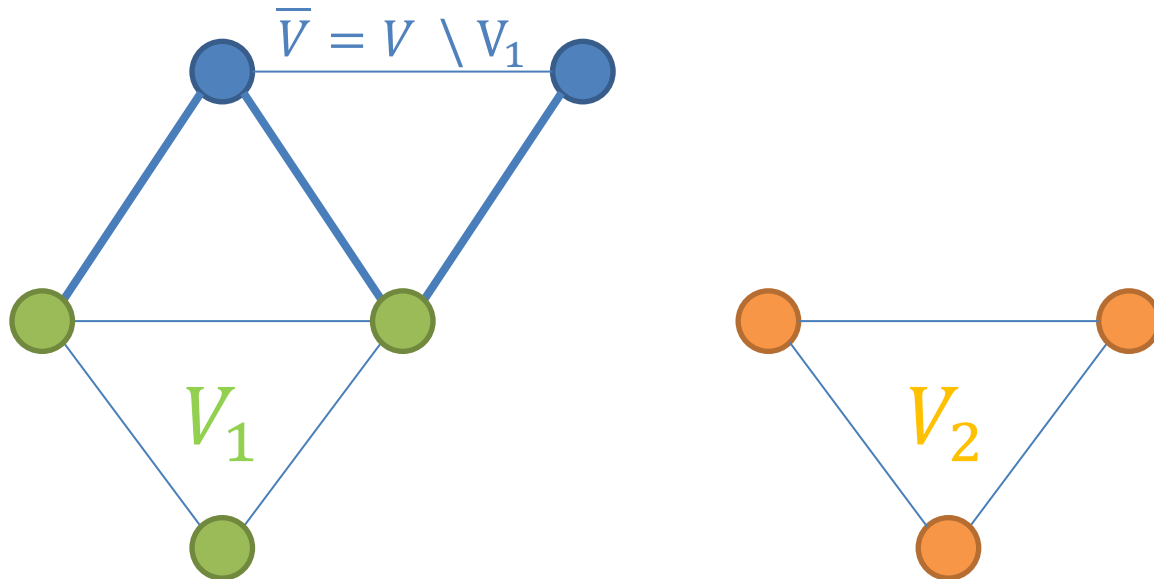
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$
- For every $(u_1, v_1) \in V_1 \times V_1$ add $(u_2, v_2) \in V_2 \times V_2$



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

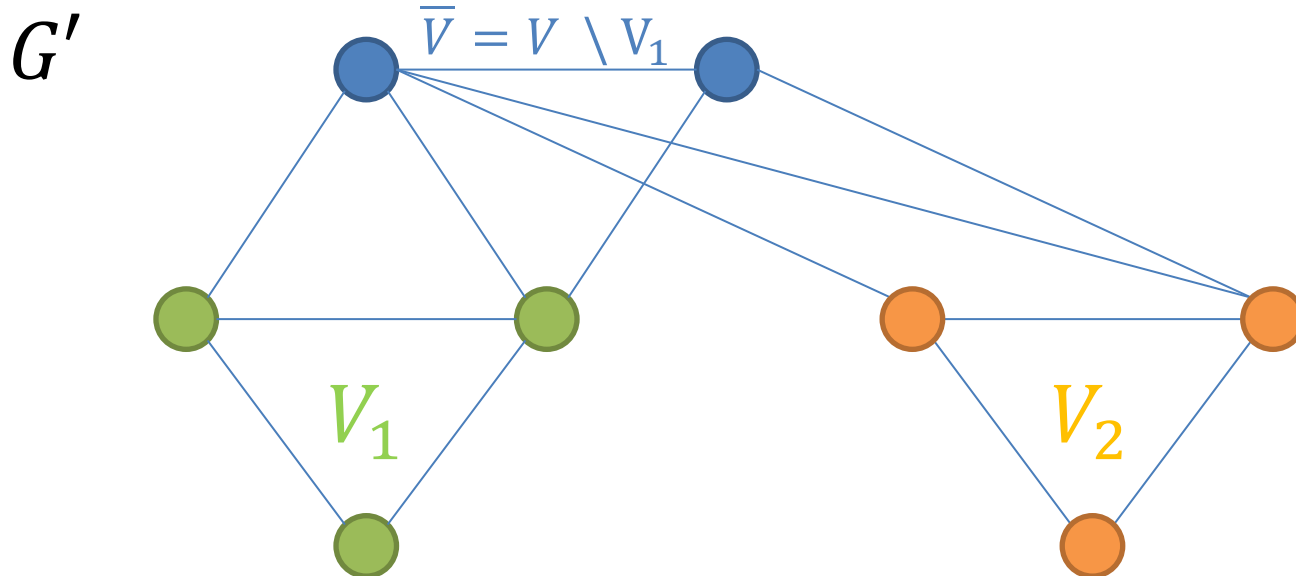
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$
- For every $(u_1, v_1) \in V_1 \times V_1$ add $(u_2, v_2) \in V_2 \times V_2$
- For every $(u_1, v) \in V_1 \times \bar{V}$, add $(u_2, v) \in V_2 \times \bar{V}$



Negative Triangle Vertex Parity (NTVP)

Reducing NWT to NTVP:

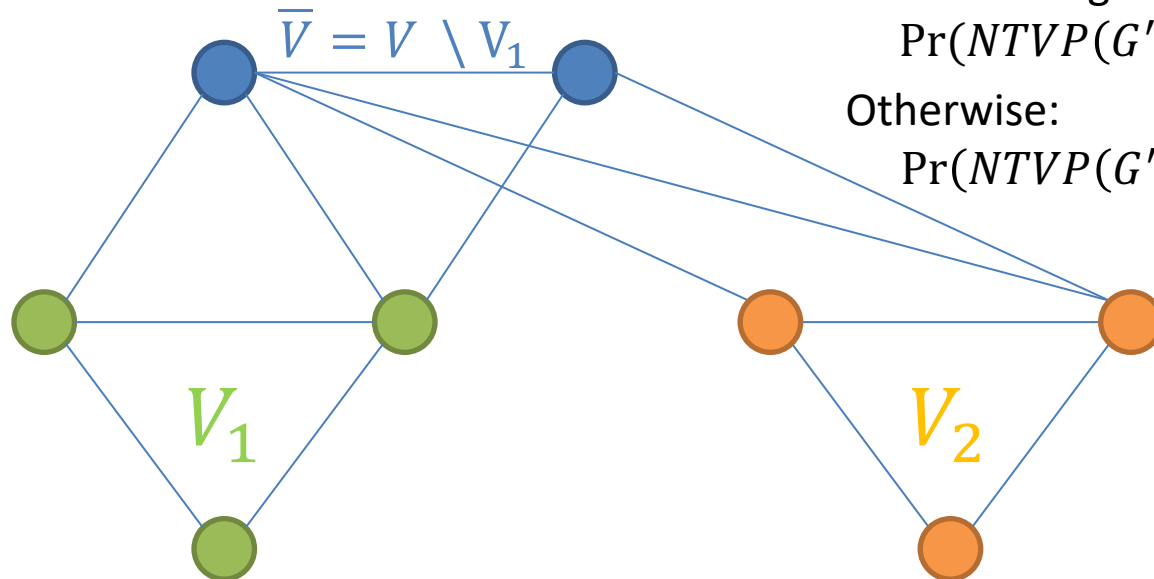
- Choose a subset of vertices $V_1 \subseteq V$ uniformly.
- For every $v_1 \in V_1$ add a copy $v_2 \in V_2$
- For every $(u_1, v_1) \in V_1 \times V_1$ add $(u_2, v_2) \in V_2 \times V_2$
- For every $(u_1, v) \in V_1 \times \bar{V}$, add $(u_2, v) \in V_2 \times \bar{V}$



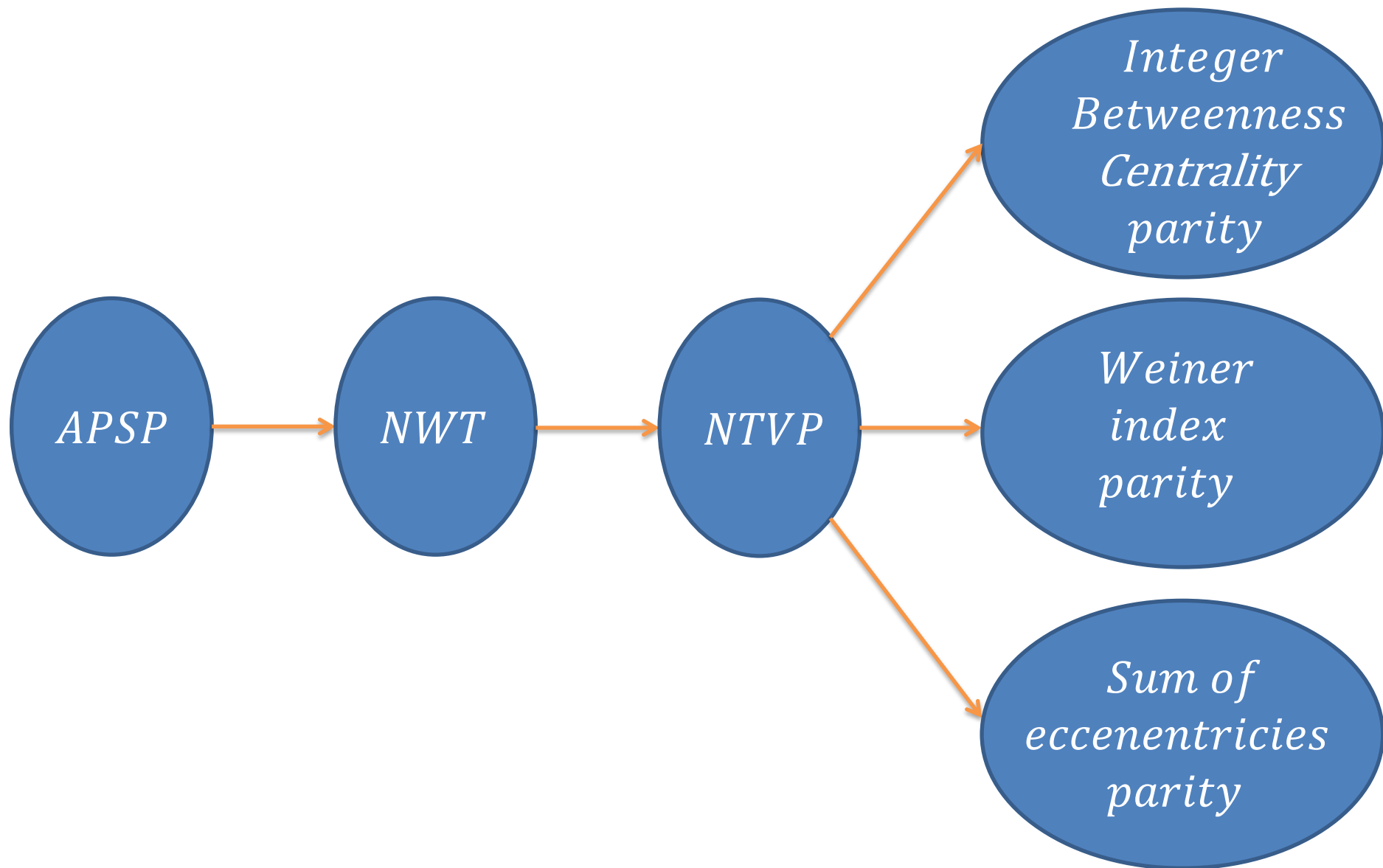
Negative Triangle Vertex Parity (NTVP)

- $v_1 \in V_1$ is in a negative triangle iff $v_2 \in V_2$ is in a negative triangle
- Vertices in V_1 and V_2 do not affect $\text{NTVP}(G')$.
- $v \in \bar{V}$ is in a negative triangle in G' iff v is in a negative triangle in G .
- $\text{NTVP}(G')$ is the NTVP of all vertices in \bar{V} .

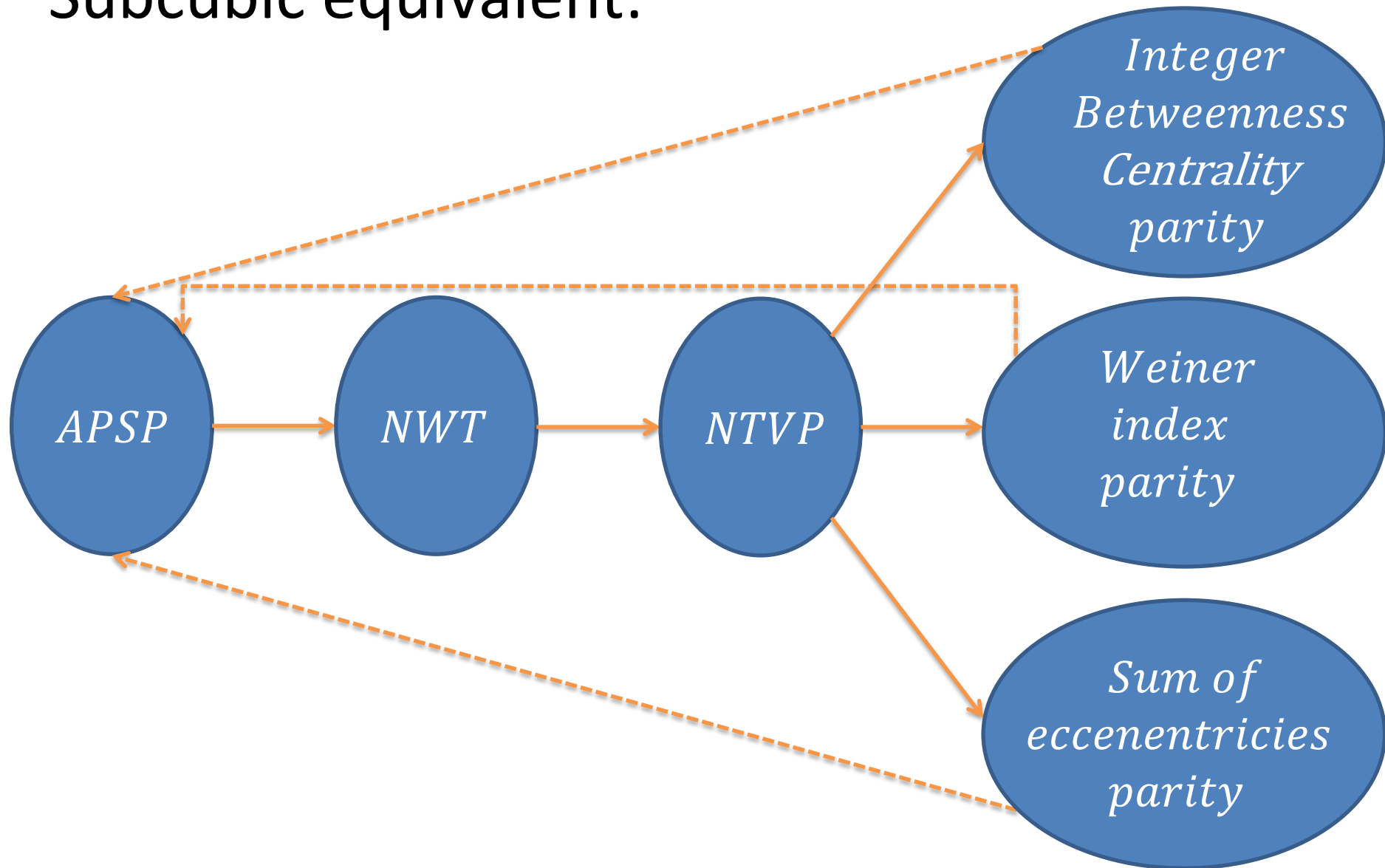
G'



If G has a negative triangle:
 $\Pr(\text{NTVP}(G') = 0)$ is 0.5
Otherwise:
 $\Pr(\text{NTVP}(G') = 0)$ is 1



Subcubic equivalent:



Negative Weight Triangle & Zero Weight Triangle

Negative Weight Triangle (NWT)

A more natural parity version of NWT:
Parity of the number of negative triangles.

APSP-equivalent:

Counting up to $O(n^{2.99})$ negative triangles. [VV Williams, R Williams '10]

Approximate counting of negative triangles.
[Dell, Lapinskas '17]

Is Negative Triangle Counting subcubic equivalent to APSP? Parity counting?

Negative Weight Triangle & Zero Weight Triangle

Negative Weight Triangle (NWT)

A more natural parity version of NWT:
Parity of the number of negative triangles.

APSP-equivalent:

Counting up to $O(n^{2.99})$ negative triangles. [VV Williams, R Williams '10]

Approximate counting of negative triangles.
[Dell, Lapinskas '17]

Is Negative Triangle Counting subcubic equivalent to APSP? Parity counting?

Zero Weight Triangle (ZWT)

Input: An edge-weighted graph G .
Output: Is there a triangle in G with total weight zero?

Reduction from 3-SUM to ZWT.
Reduction from APSP to ZWT.
[VV Williams, R Williams '13]

Solving ZWT in $O(n^{3-\epsilon}) \rightarrow$ breakthrough.

Reducing ZWT to NWT \rightarrow breakthrough.

Negative Weight Triangle & Zero Weight Triangle

Intuition: ZWT is generally harder than NWT

We show: Counting ZWTs reduces to counting NWTs

We show: Parity counting ZWTs reduces to parity counting NWTs

Negative Weight Triangle & Zero Weight Triangle

Intuition: ZWT is generally harder than NWT

We show: Counting ZWTs reduces to counting NWTs

Proof:

- (1) Compute the number of triangles in time $O(n^\omega) = O(n^{2.38})$
- (2) Compute the number of negative triangles, using an NWT counting algorithm.
- (3) Compute the number of positive triangles, using an NWT counting algorithm.
- (4) The number of zero weight triangles is (1) – (2) – (3).

Negative Weight Triangle & Zero Weight Triangle

- Counting NWTs is (conditionally) harder than NWT:
Reduction from counting NWT to NWT → breakthrough.
- In the paper:
Reduction from ZWT to ZWT parity counting.
- So there is also no reduction from NWT parity to NWT (conditioned)

The min-plus convolution class

- Min-plus convolution:

Input: Two lists $A[1..n], B[1..n]$ of n integers each.

Output: Return a list C s.t $C[k] = \min_{i+j=k} A[i] + B[j]$

The min-plus convolution class

- Min-plus convolution:
Input: Two lists $A[1..n], B[1..n]$ of n integers each.
Output: Return a list C s.t $C[k] = \min_{i+j=k} A[i] + B[j]$
- Knapsack:
Input: n items (w_i, v_i) , target weight t
Output: $\max\{\sum_{i \in I} v_i \mid \sum_{i \in I} w_i \leq t\}$
- Super-additivity:
Input: List A of n integers
Output: Check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

[Cygan et al. '17]

The following are equivalent:

$O(n^{2-\varepsilon})$ Min-plus convolution

$O((n + t)^{2-\varepsilon'})$ Knapsack

$O(n^{2-\varepsilon''})$ Super-additivity

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Knapsack is subquadratic equivalent to Knapsack Parity (and Min-plus convolution)

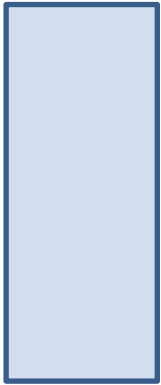
Proof: Modify original reduction [Cygan et al. '17] from Super-Additivity to Knapsack

Set $A[i] \leftarrow A[i] + W \cdot i$ for large enough W , so $0 = A[0] < A[1] < \dots < A[n - 1]$.

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight



Value



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

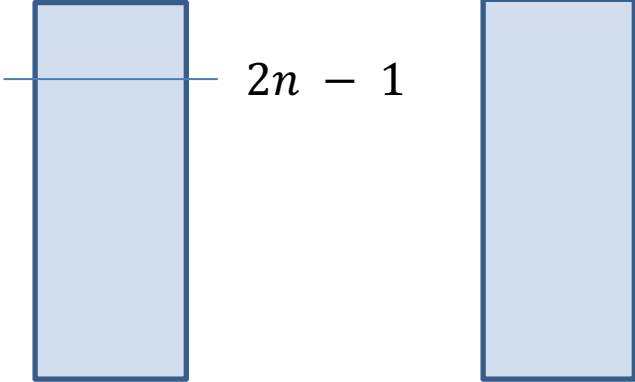
$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight

Value



$2n - 1$

Type A: $(i, 2A[i])$

Type B:

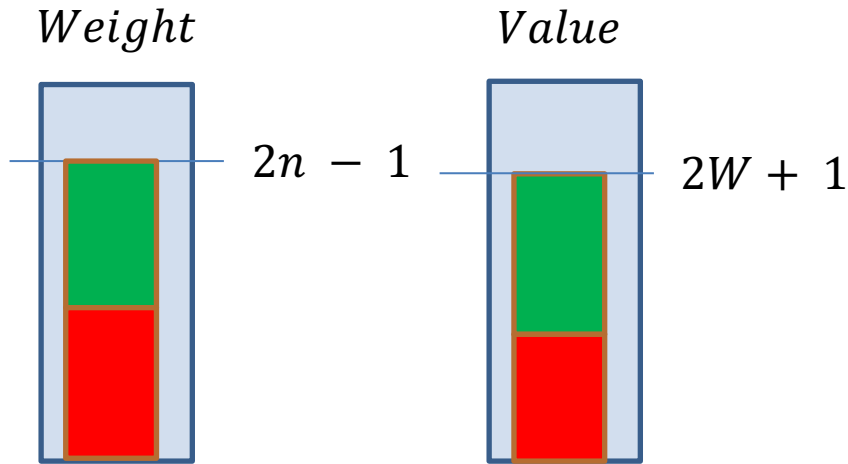
$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

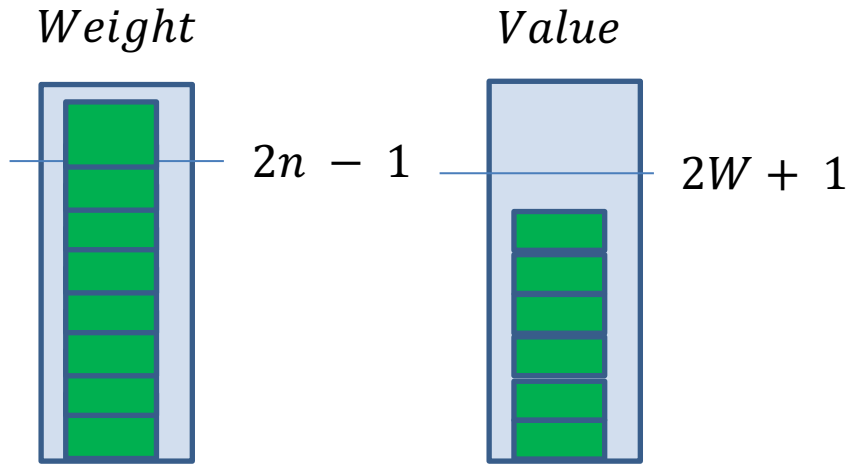
Target weight: $t = 2n - 1$

Possible solution: $(1, 2A[1])$, $(2n - 1 - 1, 2(W - A[1]) + 1)$. Value = $2W + 1$

Claim: A is super-Additive iff the solution for Knapsack is odd

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

Possible solution: $(1, 2A[1])$, $(2n - 1 - 1, 2(W - A[1]) + 1)$. Value = $2W + 1$

Claim: A is super-Additive iff the solution for Knapsack is odd

Optimal solution has **1 Type B item**:

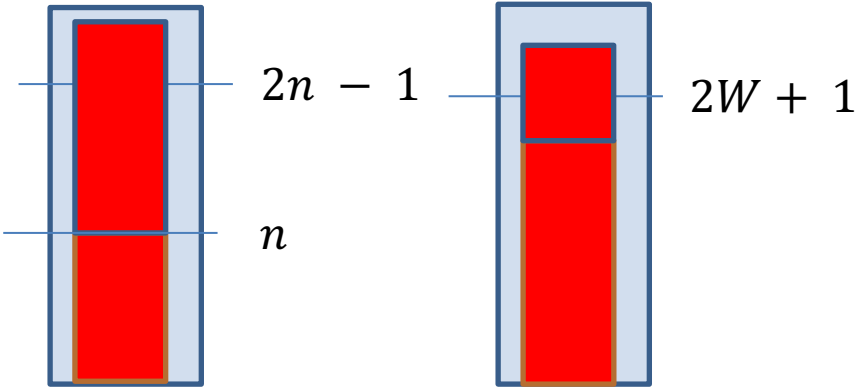
0 Type B items: Value is at most $\sum_i 2A[i] < 2W$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight

Value



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

Possible solution: $(1, 2A[1])$, $(2n - 1 - 1, 2(W - A[1]) + 1)$. Value = $2W + 1$

Claim: A is super-Additive iff the solution for Knapsack is odd

Optimal solution has **1 Type B item**:

0 Type B items: Value is at most $\sum_i 2A[i] < 2W$

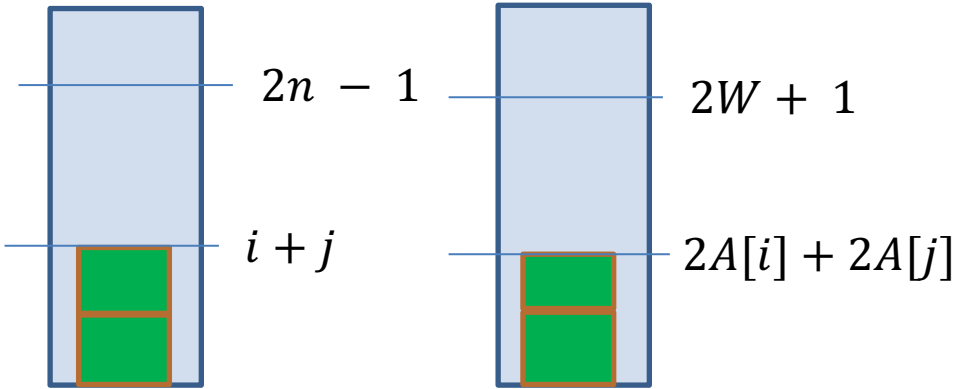
2 Type B items: Weight exceeds $2n - 1$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight

Value



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

If A is super-additive:

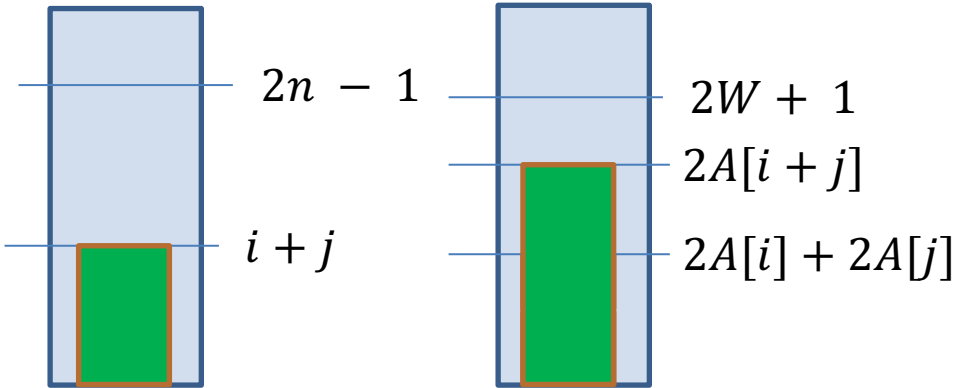
Optimal solution has **1 Type A item**: Replace $(i, 2A[i]), (j, 2A[j])$ with $(i + j, 2A[i + j])$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight

Value



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

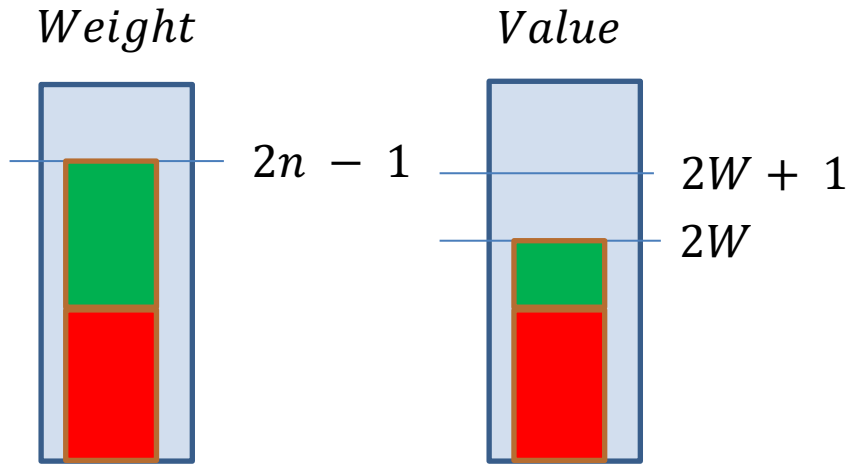
Target weight: $t = 2n - 1$

If A is super-additive:

Optimal solution has 1 Type A item: Replace $(i, 2A[i])$, $(j, 2A[j])$ with $(i + j, 2A[i + j])$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

If A is super-additive:

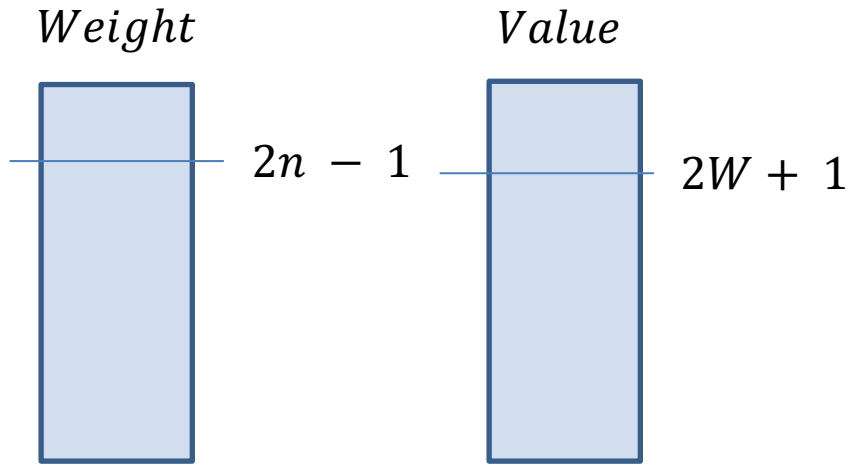
Optimal solution has **1 Type A item**: Replace $(i, 2A[i]), (j, 2A[j])$ with $(i + j, 2A[i + j])$

Optimal solution has **1 Type A item** and **1 type B item**: $(k, 2A[k]), (2n - 1 - k, 2(W - A[k]))$

Unless $k = 1 \rightarrow$ Optimal solution is odd

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

If A is **not** super-additive:

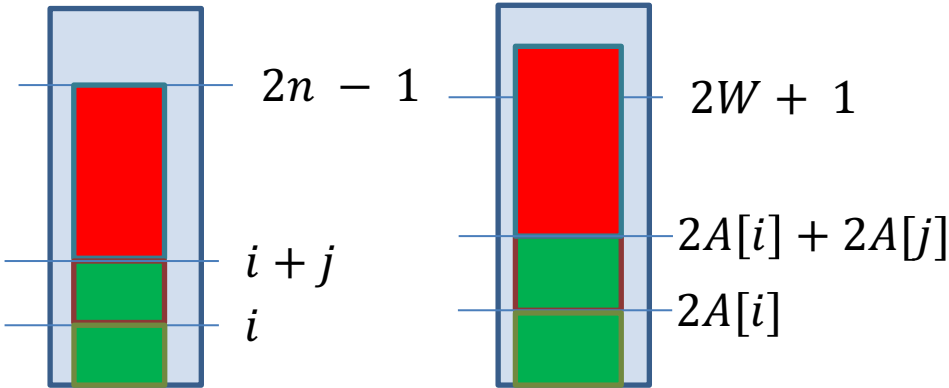
For some i, j we have $A[i] + A[j] > A[i + j]$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight

Value



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

If A is **not** super-additive:

For some i, j we have $A[i] + A[j] > A[i + j]$

Solution with value larger than $2W + 1$:

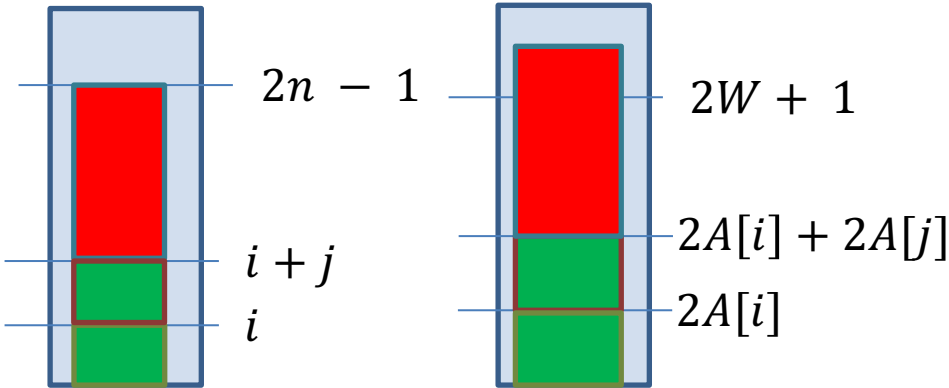
$(i, 2A[i]), (j, 2A[j]), (2n - 1 - (i + j), 2(W - A[i + j]))$

Super-Additivity to Knapsack Parity

check if $\forall i, j : A[i] + A[j] \leq A[i + j]$

Weight

Value



Type A: $(i, 2A[i])$

Type B:

$i \neq 1: (2n - 1 - i, 2(W - A[i]))$

$i = 1: (2n - 1 - 1, 2(W - A[i]) + 1)$

Target weight: $t = 2n - 1$

If A is **not** super-additive:

For some i, j we have $A[i] + A[j] > A[i + j]$

Solution with value larger than $2W + 1$:

$(i, 2A[i]), (j, 2A[j]), (2n - 1 - (i + j), 2(W - A[i + j]))$

If we use **Type B item for $i = 1$** \rightarrow Maximum value = $2W + 1$

Optimal solution does not this item \rightarrow The optimal solution is even

Problem	Complexity
APSP	Parity is Subcubic Equivalent to APSP
Median	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Radius	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Sum of Eccentricities	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Integer Betweenness Centrality	Subcubic Equivalent to APSP, Approximation is Subcubic Equivalent to Diameter under randomized reductions, Parity is Subcubic Equivalent to APSP
Second Shortest Path	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Weiner index	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP

Problem	Complexity
Maximum Subarray	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Min-plus matrix multiplication	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Replacement Paths	Subcubic Equivalent to APSP, Parity is Subcubic Equivalent to APSP
Negative Weight Triangle	Subcubic Equivalent to APSP, Approximation counting is equivalent to APSP under randomized reductions Vertex parity is Subcubic Equivalent to APSP, Randomized reductions from APSP and 3SUM to parity counting
Zero Weight Triangle	Reductions from APSP and 3SUM, Randomized reduction to parity counting and vertex parity

Problem	Complexity
Diameter	Parity is Subcubic Equivalent to Diameter
Min-plus convolution	Reductions to APSP and 3SUM, Parity is subquadratic equivalent to Min-plus convolution
0/1-Knapsack	Subquadratic equivalent to Min-plus convolution under randomized reductions Parity is subquadratic equivalent to Min-plus convolution
Reach Centrality	Subcubic Equivalent to Diameter, Parity is Subcubic Equivalent to Diameter

Problem	Complexity
Maximum Row Sum	Reduction from Co-Negative Triangle to parity
Tree sparsity	Subquadratic equivalent to Min-plus convolution under randomized reductions Parity is subquadratic equivalent to Min-plus convolution
Maximum Consecutive Subsums	Subcubic Equivalent to Diameter, Parity is Subcubic Equivalent to Diameter
Co-Negative Triangle	Reduction to Diameter, Reduction to Maximum Row Sum Parity