Version: 1.0.0 **Target:** 0.143.0-Beta

1. File: contest_tools/data_aggregators/multiplier_stats.py Baseline: Version 0.125.3-Beta

2. Surgical Changes:

- **Enhance Data Collection:** In `get_multiplier_breakdown_data`, calculate the `Run`/`S&P`/`Unk` status for every multiplier instance and store it in a `mult_status_map`.
- **Refine `build_row`:** Calculate `unique_run`, `unique_sp`, and `unique_unk` counts for each station by intersecting their "Unique" set (Log - Common) with the status map.
- **Update Return Structure:** Include these new metrics in the returned `stations_list`.

3. Surgical Change Verification (`diff`): --- BEGIN DIFF --- --- contest_tools/data_aggregators/multiplier_stats.py +++ contest_tools/data_aggregators/multiplier_stats.py @@ -127,6 +127,8 @@ # subset_keys: 'TOTAL', 'TOTAL_{Rule}', '{Band}', '{Band}_{Rule}' subsets = {call: {} for call in all_calls}

- `    # Map to store Run/S&P status for each unique multiplier instance: map[call][composi`
- `    mult_status_map = {call: {} for call in all_calls}`
- `    valid_bands = self.contest_def.valid_bands`

`    for rule in self.contest_def.multiplier_rules:`

@@ -141,6 +143,10 @@ # Filter valid valid = df[df[mult_column].notna() & (df[mult_column] != 'Unknown')]

- `            # Pre-calculate status for all multipliers in this scope`
- `            # Status logic: If worked on Run, it's Run. If Mixed, it's Mixed.`
- `            status_series = valid.groupby(mult_column)['Run'].apply(self._get_run_sp_sta`
- `            if method == 'once_per_log':`
  `    # Scope: Global. Key: (Rule, Value)`
  `    # Just grab unique values`

@@ -148,6 +154,8 @@ for val in uniques: composite_key = (mult_name, 'All', val) raw_sets[call].add(composite_key)

- `                    # Store status`
- `                    mult_status_map[call][composite_key] = status_series.get(val, 'Unk')`

`                    # Populate Subsets`
`                    # 1. Total`

@@ -161,6 +169,7 @@ grouped = valid.groupby('Band')[mult_column].unique() for band, values in grouped.items(): for val in values: composite_key = (mult_name, band, val) raw_sets[call].add(composite_key)

- `                        # Note: status_series is global for the log, which is fine for '`
- `                        # Refinement: Get status specific to this band if possible, but`
- `                        # Better: Re-run group by for status per band to be precise`
- `                        # Optimization: Just look it up from the valid df subset for thi`
- `                        # For now, using global lookup is consistent with 'valid' df sco`
- `                        mult_status_map[call][composite_key] = status_series.get(val, 'U`
- `                        # Populate Subsets`
- `                        # 1. Total (Global Score Sum)`

@@ -183,9 +192,27 @@ stations_list = [] for call in all_calls: metrics = comp.station_metrics.get(call) +

- `            # Calculate Unique Breakdown`
- `            unique_run = 0`
- `            unique_sp = 0`
- `            unique_unk = 0`
- `            `
- `            if metrics:`
- `                # Unique = Log Set - Common Set`
- `                log_set = sets_to_compare[call]`
- `                unique_items = log_set - comp.common_set if hasattr(comp, 'common_set')`
- `                # Recalculate common for safety since ComparativeEngine returns counts`
- `                # Actually, common is intersection of all.`
- `                # Let's just use the logic: Unique = (Log Count - Common Count) is just`
- `                # We need the ACTUAL items.`
- `                # ComparativeEngine doesn't expose the Common Set publicly in StationMet`
- `                # We must re-derive 'Common' locally or upgrade Engine.`
- `                # Local derivation:`
- `                common_items = set.intersection(*sets_to_compare.values()) if sets_to_co`
- `                unique_items = sets_to_compare[call] - common_items`
- `                `

```
                for item in unique_items:
                    status = mult_status_map[call].get(item, 'Unk')
                    if status == 'Run': unique_run += 1
                    elif status == 'S&P': unique_sp += 1
                    else: unique_unk += 1
            if metrics:
                stations_list.append({
                    'count': metrics.count,
                    'delta': metrics.count - comp.universe_count # Delta from Par (Total
                    'delta': metrics.count - comp.universe_count, # Delta from Par (Tota
                    'unique_run': unique_run,
                    'unique_sp': unique_sp,
                    'unique_unk': unique_unk
                })
            else:
                stations_list.append({'count': 0, 'delta': -comp.universe_count})
                stations_list.append({
                    'count': 0,
                    'delta': -comp.universe_count,
                    'unique_run': 0,
                    'unique_sp': 0,
                    'unique_unk': 0
                })

        return {
            'label': label,
```

--- END DIFF ---

4. File: contest_tools/reports/text_multiplier_breakdown.py Baseline: Version 0.142.0-Beta

5. Surgical Changes:

- **Update `format_row`:** Update the row formatter to accept the new `unique_run`, `unique_sp`, `unique_unk` keys and format them as `[Run:XX S&P:YY Unk:ZZ]`.
- **Widen Columns:** Increase the column width calculation to accommodate the verbose format (~30 chars).

6. Surgical Change Verification (`diff`): --- BEGIN DIFF --- --- contest_tools/reports/text_multiplier_breakdown.py +++ contest_tools/reports/text_multiplier_breakdown.p @@ -52,14 +52,17 @@ # Helper to format a row

- ```
  # Layout: Scope (25), Total (8), Common (8), Call1 (15), Call2 (15)...
  ```
- ```
  # Layout: Scope (25), Total (8), Common (8), Call1 (30), Call2 (30)...
  def format_row(label, total, common, station_dict, indent=0):
      prefix = " " * (indent * 2)
      lbl = f"{prefix}{label}"

      # Base columns
      row_str = f"{lbl:<25} {total:>8} {common:>8}"

      # Station columns
      # Re-implement row formatter to handle list input correctly
      # The previous code assumed station_dict was a dict.
      for i, call in enumerate(all_calls):
          stats = station_dict[i] # station_dict is actually a list here
  ```
- ```
          # Fixed-width formatting: Count(6) + Spacer(1) + Delta(7) = 14 chars
  ```
- ```
          count_part = f"{stats['count']:>6}"
  ```
- ```
          # Verbose Breakdown: "972 [Run:60 S&P:25 Unk:5] -39"
  ```
- ```
          count_part = f"{stats['count']}"
  ```
-
- ```
          u_run = stats.get('unique_run', 0)
  ```
- ```
          u_sp = stats.get('unique_sp', 0)
  ```
- ```
          u_unk = stats.get('unique_unk', 0)
  ```
-
- ```
          # Only show breakdown if there are uniques
  ```
- ```
          total_unique = u_run + u_sp + u_unk
  ```
- ```
          if total_unique > 0:
  ```
- ```
              breakdown = f"[Run:{u_run} S&P:{u_sp} Unk:{u_unk}]"
  ```
- ```
          else:
  ```
- ```
              breakdown = "[Par]"
  ```
- ```
          delta = stats.get('delta', 0)
          delta_part = f"({delta})" if delta < 0 else "0"
  ```
- ```
          row_str += f" {count_part} {delta_part:<7}"
  ```
- ```
          # Align: Count (5) + Breakdown (22) + Delta (6) ~= 33 chars
  ```

```
            col_text = f"{count_part:>5} {breakdown:<24} {delta_part:>6}"
            row_str += f" {col_text}"
        return row_str

    # Header Row
    header = f"{'Scope':<25} {'Total':>8} {'Common':>8}"
    for call in all_calls:
        # Center alignment (14 chars) + 1 leading space = 15 chars total per column
        header += f" {call:^14}"
        # Center alignment (~35 chars)
        header += f" {call:^37}"

    table_width = len(header)
```

--- END DIFF ---

7. File: web_app/analyzer/templates/analyzer/multiplier_dashboard.html
Baseline: Version 0.125.1-Beta

8. Surgical Changes:

- **Redesign Summary Card:** Implement "Concept 7" (Integrated Data Rows) for the top summary card.
- **Redesign Grid:** Switch from 2-Column to 3-Column grid (2 rows for 6 bands).
- **Implement Data Rows:** Replace the simple tables with the `band-row` divs containing the stacked bar and the 3 numeric columns.

9. Surgical Change Verification (`diff`): --- BEGIN DIFF --- --- web_app/analyzer/templates/analyzer/multiplier_dashboard.html +++ web_app/analyzer/templates/analyzer/multiplier_dashboard.html @@ -128,11 +128,49 @@

```
        <span class="fw-bold text-secondary"><i class="bi bi-grid-3x3-gap me-2"></i>Mult
        <span class="fw-bold text-secondary"><i class="bi bi-grid-3x3-gap me-2"></i>Mult
        <div class="d-flex gap-3 text-muted small">
            <div class="d-flex align-items-center"><div style="width:8px;height:8px;back
            <div class="d-flex align-items-center"><div style="width:8px;height:8px;back
            <div class="d-flex align-items-center"><div style="width:8px;height:8px;back
            <div class="d-flex align-items-center" style="border-left:1px solid #ddd; pa
        </div>
        <div>
            {% if breakdown_txt_url %}
```

@@ -144,30 +182,75 @@ {{ full_contest_title }}

- <p class="text-muted small">Multiplier Analysis • {{ all_calls|join:",
- <p class="text-muted small">Group Par Analysis • {{ all_calls|join:", "

- <table class="table table-sm table-hover mb-0 text-center align-middle borde
- <thead class="table-dark">
- <tr>
- <th class="text-start ps-3" style="width: 20%;">Scope</th>
- <th>Total Worked</th>
- <th>Common</th>
- {% for call in all_calls %}
- <th>{{ call }}</th>
- {% endfor %}
- </tr>
- </thead>
- <tbody>
- {% include "analyzer/partials/breakdown_rows.html" with rows=breakdo
- </tbody>
- </table>
- +
-
- <h6 class="m-0 fw-bold">Tournament Summary (All Bands)</h6>
-
-
- <div class="bg-light fw-bold text-muted text-uppercase small p-2 d-f
- Station
- Base
- Unique Composition
- R</div
- S</div

```
                        <div style="width:50px;" class="text-center text-secondary">U</d
                        <div style="width:60px;" class="text-end text-danger">Gap</div>
                    </div>
                    {% for row in breakdown_totals %}
                        {% for stat in row.stations %}
                        +                              +                              {% wit
                        <div class="d-flex align-items-center p-2 border-bottom" style="
                            <div style="width:100px;" class="fw-bold">{{ all_calls|get_i
                            <div style="width:80px;" class="text-center text-muted fw-bo
                            <div class="flex-grow-1 px-3">
                                <div style="height:20px; background:#fff1f2; border:1px
                                    <div style="height:100%; background:#22c55e; width:{
                                    <div style="height:100%; background:#3b82f6; width:{
                                    <div style="height:100%; background:#9ca3af; width:{
                                </div>
                            </div>
                            <div style="width:50px;" class="text-center fw-bold text-suc
                            <div style="width:50px;" class="text-center fw-bold text-pri
                            <div style="width:50px;" class="text-center text-secondary">
                            <div style="width:60px;" class="text-end fw-bold text-danger
                        </div>
                        {% endwith %}
                        {% endfor %}
                    {% endfor %}
                </div>
            </div>
        </div>
        <div class="row g-4">
            <div class="col-lg-6">
                <h6 class="fw-bold text-secondary mb-2 border-bottom pb-2">Low Bands (16
                {% for block in low_bands_data %}
```

```
                    <div class="table-responsive mb-3 border rounded">
                        <table class="table table-sm table-hover mb-0 text-center align-
                            <thead class="table-light text-muted small">
                                <tr>
                                    <th class="text-start ps-2" style="width: 20%;">{{ b
                                    <th>Worked</th>
                                    <th>Common</th>
                                    {% for call in all_calls %}<th>{{ call }}</th>{% end
                                </tr>
                            </thead>
                            <tbody>
                                {% include "analyzer/partials/breakdown_rows.html" with
                            </tbody>
                        </table>
                    </div>
                {% endfor %}
            </div>
        +            <h6 class="text-uppercase text-muted fw-bold mb-3 ls-1">Band Perfor
        <div class="row g-4">
            +                {% for block in low_bands_data|add:high_bands_data %}
            <div class="col-lg-4">
                <div class="card h-100 shadow-sm border-top-4 border-secondary">
                    <div class="card-header d-flex justify-content-between align-items-c
                        <span class="fw-bold">{{ block.label }}</span>
                        {% with first_row=block.rows.0 %}
                        <span class="badge bg-light text-dark border">Par: {{ first_row.
                        {% endwith %}
                    </div>
                    <div class="card-body p-0">
                        <div class="bg-light text-muted p-2 d-flex small fw-bold border-
                            <div style="width:45px;">Call</div>
```

```
                            <div class="flex-grow-1 text-center">Unique</div>
                            <div style="width:25px;" class="text-center">R</div>
                            <div style="width:25px;" class="text-center">S</div>
                            <div style="width:25px;" class="text-center">U</div>
                            <div style="width:35px;" class="text-end">Gap</div>
                        </div>
                        {% for row in block.rows %}
                            {% for stat in row.stations %}
                            {% with total_unique=stat.unique_run|add:stat.unique_sp|add:
                            <div class="d-flex align-items-center p-2 border-bottom" sty
                                <div style="width:45px;" class="fw-bold">{{ all_calls|ge
                                <div class="flex-grow-1 px-1">
                                    <div style="height:12px; background:#fff1f2; border:
                                        <div style="height:100%; background:#22c55e; wid
                                        <div style="height:100%; background:#3b82f6; wid
                                        <div style="height:100%; background:#9ca3af; wid
                                    </div>
                                </div>
                                <div style="width:25px;" class="text-center text-success
                                <div style="width:25px;" class="text-center text-primary
                                <div style="width:25px;" class="text-center text-seconda
                                <div style="width:35px;" class="text-end fw-bold text-da
                            </div>
                            {% endwith %}
                            {% endfor %}
                        {% endfor %}
                    </div>
                </div>
            </div>
            {% endfor %}
        </div>
```

--- END DIFF ---

10. Affected Modules Checklist:

- `contest_tools/data_aggregators/multiplier_stats.py`
- `contest_tools/reports/text_multiplier_breakdown.py`
- `web_app/analyzer/templates/analyzer/multiplier_dashboard.html`

11. Pre-Flight Check:

- **Inputs:** Existing logs.
- **Expected Outcome:** * Text report will show `[Run:60 S&P:25 Unk:5]` breakdown.
  - Dashboard will show Full-Width Summary + 3x2 Band Grid with green/blue bars and numeric columns.
- **Mental Walkthrough:** 1. Aggregator loop runs. `status_series` calculates mode for each mult. 2. `mult_status_map` populated. 3. `build_row` iterates unique items, looks up status, increments `unique_run/sp/unk`. 4. Text report formatter widens columns and prints verbose string. 5. Web template iterates `low_bands + high_bands` list to render 3-column grid.

12. Post-Generation Verification:

- Confirm text report format.
- Confirm web dashboard layout (3 columns).
- Confirm Green/Blue/Gray bars appear.