For example, one can replace the object *Man* by object *Woman* from the library [scene (c)]. or components of the MPEG-4 video object encoder concurrent playout of VO₀ and VO₁, followed by a object VO₃ at the time tᵢ. Here, VO₂ and AO₀ can i as the background audio-visual object throughout i
Fig. 2. Playout scenario of an MPEG-4 multimedia session.
Fig. 3. Proposed MPEG-4 system encoder implementation.
Modeling MPEG-4 Interactive

MOCPN--- An extension of Object Composition Petri Net

MOCPN=\{P, T,A,M,D,TS,PT\}

Run time MOCPN generation according to the requirement of live video: mostly caused by user interaction. Such as old video cease and new video inserted.
Fig. 4. Example of hierarchical MOCPN.
Three Scheduling Algorithms

Goal: Distribute the encoding task of multiple video object to the cluster workstations for high speed up.

Need a dynamic scheduling scheme because the non static nature of task situation.

Scheduling is NP in nature, need to find heuristic methods

RRS (Round Robin Scheduling)

GS (Group Scheduling)

GAS (GOV adjusting scheduling)
**Notation and Equation used**

- $V_{o_j}$: $i$th video object
- $V_{o_{p_{i,j}}}$: $j$th frame of video object $V_{o_j}$
- $\delta_j$: start time of the video object $V_{o_j}$
- $\tau_i$: synchronization interval of the video object $V_{o_j}$
- $S_{j,j}$: Size of $V_{o_{p_{i,j}}}$
- $d_{j,i}$: Playout deadline of $V_{o_{p_{i,j}}}$
- $T_{v_s}$: Total execution time for entire session
- $T_{v_{o_j}}(i)$: execution time cost for $V_{o_j}$
- $T_{v_{o_{p_{i,j}}}}(i,j)$: execute time for $V_{o_{p_{i,j}}}$
- $T_{s}$: Scheduling time for task assignment
Additional Notation for Scheduling Algorithms

\( \mathcal{V} \) Scheduling sequence of the video session

\( P_k \) kth workstation of the system. \( K=1..K \)

\( R_k \) scheduler on \( P_k \)

\( L_x \) scheduling intervals triggered by user interaction

\( V_x \) partitioned data of the VOP to \( P_k \)
$T_{enc}$ Encode time.
$T_{com}$ Interprocessors Communication time.
$T_p$ data partitione time.

$T_{vop}(i,j) = \max_k \{ T_{enc}^k(i,j) + T_{com} + T_p \}$

$T_{enc}^k(i,j) = \alpha \cdot (S_{i,j}/K)$

$T_{vo}(i) = T_s + \sum_j (\max_k \{ T_{enc}^k(i,j) + T_{com} + T_p \})$

To preserve the intraobject synchronization the encoding time must satisfy the following constraint:

$\sum_j (\max_k \{ T_{enc}^k(i,j) + T_{com} + T_p \}) < d_{ij} - \delta_{ij}$
RRS:
Initialize the scheduling interval \( L_x \), where \( x=0 \);
Sort all VOP in \( v \) using EDF(earliest deadline first) rule.
Initialize \( R_k = {} \) and \( g_0 = K \), where \( P_k \in G_0 \), \( k=0 \ldots K-1 \);
Point to the first VOP in \( V \);
Schedule \( R_k \) to the current VOP:
\[ R_k = R_k \cup \{VOP\} \]
Partition the VOP and map the data area \( v_k \) to \( P_k \),
Go to next VOP if no user event, else trigger the scheduler \( R_k \) to update the state of the video objects and begin the next scheduling interval \( L_x \), where \( x=x+1 \), go to step 2 repeat the last three step.
GS Algorithms: to minimize the interprocessors communication posed in RSS Workstations was divided into N groups corresponding N independent video objects.

The number of workstations in each group is proportional to the size and play out duration

the GS only determines the workstation assignment at the beginning of each scheduling interval Lx.

\[ \eta_i = K \cdot \frac{S_{x,i}}{\tau_{x,i}} / \sum_{i=0}^{N-1} \frac{S_{x,i}}{\tau_{x,i}} \]

\[ g_i = \text{the workstation number for ith group, which is the floor of } \eta_i \]
Fig. 6. MOCPN model for RRS scheduling.

Fig. 7. MOCPN model for GS scheduling.
<table>
<thead>
<tr>
<th>Video object</th>
<th>Size ratio</th>
<th>4 workstations</th>
<th>8 workstations</th>
<th>12 workstations</th>
<th>16 workstations</th>
<th>20 workstations</th>
</tr>
</thead>
<tbody>
<tr>
<td>VO₀</td>
<td>0.7</td>
<td>1(25%)</td>
<td>5(62%)</td>
<td>8(67%)</td>
<td>11(69%)</td>
<td>14(70%)</td>
</tr>
<tr>
<td>VO₁</td>
<td>0.2</td>
<td>1(25%)</td>
<td>1(12.5%)</td>
<td>2(17%)</td>
<td>3(19%)</td>
<td>4(20%)</td>
</tr>
<tr>
<td>VO₂</td>
<td>0.05</td>
<td>1(25%)</td>
<td>1(12.5%)</td>
<td>1(12.5%)</td>
<td>1(6%)</td>
<td>1(5%)</td>
</tr>
<tr>
<td>VO₃</td>
<td>0.05</td>
<td>1(25%)</td>
<td>1(12.5%)</td>
<td>1(12.5%)</td>
<td>1(6%)</td>
<td>1(5%)</td>
</tr>
</tbody>
</table>
GAS Algorithms To overcome the load balance and limitation of workstations number in GS
GAS periodically detect the workload of the workstations and performs rescheduling
Smallest Object merged into one group to overcome the load imbalance, Merge stop criteria: the following inequation which indicate the imbalance did not hold.
\[ \alpha \cdot \left( \frac{S_{x,0}}{\tau_{x,0}} \right) / \sum_{i=0}^{N-1} \left( \frac{S_{x,i}}{\tau_{x,i}} \right) < \frac{1}{\beta \cdot K} \]
### Table III
**Object Merging Procedure (Workstations = 4)**

<table>
<thead>
<tr>
<th>Video object</th>
<th>Size ratio</th>
<th>Workstations</th>
<th>Condition in Equation (9)</th>
<th>1st merging</th>
<th>Condition in Equation (9)</th>
<th>2nd merging</th>
<th>Condition in Equation (9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VO₀</td>
<td>0.7</td>
<td>1(25%)</td>
<td></td>
<td>0.7</td>
<td>2(50%)</td>
<td>0.7</td>
<td>3(75%)</td>
</tr>
<tr>
<td>VO₁</td>
<td>0.2</td>
<td>1(25%)</td>
<td>0.05&lt;0.125 continue</td>
<td>0.2</td>
<td>1(25%)</td>
<td>0.3</td>
<td>1(25%) stop</td>
</tr>
<tr>
<td>VO₂</td>
<td>0.05</td>
<td>1(25%)</td>
<td></td>
<td>0.1</td>
<td>1(25%)</td>
<td>0.3</td>
<td>1(25%) stop</td>
</tr>
<tr>
<td>VO₃</td>
<td>0.05</td>
<td>1(25%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 8. MOCPN model of GAS algorithm.
Fig. 9. Size variation of VO₀ “Akiyo” and VO₁ “Weather”. (a) QCIF format and (b) CIF format.